FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Sistema didático de uma unidade industrial de processos discretos

Daniel Filipe de Azeredo Silva

Mestrado Integrado em Engenharia Electrotécnica e de Computadores

Orientador: Paulo Portugal

25 de julho de 2018

Resumo

Os sistemas didáticos, que têm como principal objetivo a resolução de desafios semelhantes aos que existem no contexto real, são um recurso amplamente utilizado no contexto prático do ensino, permitindo simular o funcionamento de um sistema real na segurança de um laboratório.

Esta dissertação é baseada no sistema didático de uma unidade industrial de processos discretos, existente no Laboratório de Automação Industrial no Departamento de Engenharia Eletrotécnica e de Computadores (DEEC) da Faculdade de Engenharia da Universidade do Porto (FEUP).

Inicialmente é realizado um estudo de diferentes soluções de sistemas didáticos presentes no mercado e a forma como se podem dividir quanto à finalidade, tangibilidade e forma de controlo.

Após o estudo das soluções existentes, é feita a analise das especificações e arquitetura do sistema didático, de forma a identificar pontos de melhoria a partir dos quais é proposta e implementada uma nova arquitetura. A dissertação termina com as conclusões obtidas a partir dos resultados da implementação.

Palavras-chave — Sistema didático, automação industrial, Ensino, Industria, *softPLC*, braço robótico.

Abstract

The main objective of the didactic systems is to solve similar challenges to those that exist in the real context. They are a resource widely used in the practical context of teaching, allowing to simulate the functioning of a real system in the safety of a laboratory.

This dissertation is based on the didactic system of an industrial unit of discrete processes, existing in the Laboratory of Industrial Automation in the Department of Electrical and Computer Engineering (DEEC) of the Faculty of Engineering of the University of Porto (FEUP).

Initially a study of different solutions of didactic systems present in the market and the way in which they can be divided as to the purpose, tangibility and form of control is carried out.

After the study of the existing solutions, it is made the analysis of the specifications and architecture of the didactic system, in order to identify points of improvement from which a new architecture is proposed and implemented. The dissertation ends with the conclusions drawn from the results of the implementation.

Keywords — Didactic system, Industrial automation, Teaching, Industry, softPLC, Robotic Arm.

Agradecimentos

Agradeço aos meus pais, irmã, namorada e amigos pelo apoio que me deram, pela paciência e compreensão que demonstraram em todos os momentos menos bons.

Quero, também, agradecer ao meu orientador, Prof. Doutor Paulo Portugal, por todo o apoio dado ao longo da dissertação.

Quero, por fim, agradecer ao Departamento de Engenharia Eletrotécnica e de Computadores por me proporcionar condições para desenvolver esta dissertação.

Daniel Filipe de Azeredo Silva

"Normal people believe that if it ain't broke, don't fix it.
Engineers believe that if it ain't broke, it doesn't have enough features yet"

Scott Adams

Índice

Lista de Figuras

Lista de Tabelas

Abreviaturas e Símbolos

Abreviaturas

C.I. circuito integrado

DEEC Departamento de Engenharia Eletrotécnica e de Computadores

FB *Function Block*

FC *Function Code*

FEUP Faculdade de Engenharia da Universidade do Porto

FMS *Flexible Manufacturing System*

GRAFCET **GRA**phe **F**onctionnel de **C**ommande **É**tape/**T**ransition

HID *Human Interface Device*

HMI *Human Machine Interface*

IP *Internet Protocol*

ISP *In-system programming*

LED *Light-emitting diode*

OPC *Open Platform Communications*

PLC Controlador Lógico Programável

POU *Program Organization Unit*

PWM *Pulse Width Modulation*

RFID *Radio-frequency identification*

RGB *red, green, and blue color model*

RTU *Remote Terminal Unit*

SBC *Embedded Single Board Computers*

SCADA *Supervisory Control And Data Acquisition*

SFC *Sequential Function Chart*

S.O. Sistema Operativo

TCP *Transmission Control Protocol*

Capítulo 1

Introdução

Ao longo do percurso formativo académico de um aluno é habitual que, inicialmente, lhe sejam apresentados métodos de aprendizagem genéricos e com um elevado grau de abstração que, com o avançar do percurso, se tornam cada vez mais específicos com a área de especialização.

Neste percurso é importante uma componente prática, pois permite consolidar os conhecimentos teóricos e aplicá-los na resolução de problemas reais. Esta complementaridade de formação é fundamental para o bom desempenho profissional de um engenheiro, pois irá permitir que enfrente os desafios com segurança e capacidade de análise, de modo a encontrar e implementar propostas de solução. No entanto, a utilização de sistemas reais em ambiente académico nem sempre é viável devido a vários fatores como a dimensão do espaço, características técnicas, requisitos elétricos, condições de segurança e o custo associado a este tipo de montagem. Assim e para ultrapassar estas condicionantes, recorre-se a sistemas didáticos, que se revelam uma boa alternativa, desde que sejam desenvolvidos de forma a terem um funcionamento semelhante aos sistemas reais. Durante a sua formação, deve ser dada a conhecer aos estudantes diferentes opções existentes no mercado e dar-lhes bases para que possam saber selecionar as melhores soluções para cada problema, assim como as utilizar em contexto prático.

O sistema didático do Laboratório de Automação Industrial do Departamento de Engenharia Eletrotécnica e de Computadores da Faculdade de Engenharia da Universidade do Porto é um recurso bastante utilizado nas aulas laboratoriais de Automação Industrial, simulando uma linha de produção industrial e os problemas que seriam encontrados no sistema real.

1.1 Motivação

O motivo que me levou a interessar-me por este tema de dissertação foi a minha ligação estreita ao Laboratório de Automação Industrial do DEEC. Exercendo funções de técnico de apoio ao ensino e investigação neste laboratório, adquiri sensibilidade e competências que me levaram a questionar sobre possíveis melhorias de funcionamento e resultados deste sistema didático, encontrando aí um desafio que resultou nesta dissertação.

1.2 Objetivos

Os objetivos desta dissertação passam por estudar os sistemas didáticos no ensino da automação industrial e o sistema didático em que esta dissertação se baseia de forma a serem propostas soluções para melhorar o sistema didático e as implementar.

1.3 Estrutura do documento

Este documento divide-se em seis capítulos. No primeiro é feita a introdução da dissertação.

No segundo são descrito métodos de ensino de automação industrial e como os sistemas podem ser classificados sendo feita uma apresentação de algumas soluções existentes.

No terceiro capitulo é feita uma exposição das especificações do sistema didático sobre o qual a dissertação é realizada, tanto do seu estado inicial como depois das alterações já realizadas.

No quarto capitulo são apresentadas as soluções propostas, divididas entre nível lógico e nível físico, antecedidas do levantamento de requisitos em que foram baseadas.

No quinto capitulo é descrito, para cada solução proposta, o estudo das opções consideradas para a implementação, a justificação da opção escolhida e como foi implementada, finalizando com os resultados obtidos.

O sexto e último capítulo é composto pelas conclusões do trabalho desenvolvido e aponta para o trabalho que ainda pode ser realizado dentro do contexto desta dissertação.

Capítulo 2

Sistemas didáticos no ensino da automação industrial

2.1 Introdução

Este capitulo é divido em três partes. Numa primeira parte é feita uma descrição de métodos aplicados no ensino de automação industrial, mencionadas vantagens e desvantagens de cada método. Na segunda parte é explicado como os diferentes sistemas de automação podem ser divididos e na terceira, e ultima parte, são apresentadas algumas soluções atuais existentes.

2.2 Ensino de automação industrial

Em engenharia a utilização de laboratórios é fundamental para a formação de futuros engenheiros [16, 31]. É graças às experiências realizadas que os alunos são capazes de consolidar conhecimentos e ter contacto com situações que os preparam para a vida profissional. Neste contexto, as experiências e os laboratórios podem ser divididos como:

- Experiências físicas
  - Realizada com recurso a equipamento físico que poderá ser o mesmo que se encontra fora do mundo académico ou versões didáticas com o mesmo princípio de funcionamento.

    Vantagens: dar a conhecer equipamento físico que é utilizado fora do mundo académico e ser possível observar fenómenos que numa experiência simulada passariam despercebidos.

    Desvantagens: poderem ser danificados em situações de má utilização e não serem isentos de riscos para o utilizador.
- Experiências simuladas

  - Realizada com recurso a um programa que simula o princípio de funcionamento do equipamento físico.

    Vantagens: necessitar de poucos recursos, bastando a base computacional onde o sistema virtual é executado, e não havendo o risco de serem provocados danos a equipamento real.

    Desvantagens: abstração do sistema real e a dependência da qualidade dos modelos da simulação levam a que este meio deva substituir completamente o contacto com experiências físicas [30].

- Laboratório local

  - Estes laboratórios são de utilização presencial onde o aluno tem contacto direto com a experiência a executar, seja ela física ou simulada.

    Vantagens: contacto direto com o equipamento e a possibilidade de realizar experiências que não estão previamente definidas.

    Desvantagens: acesso aos laboratórios ser restrito e limitado à sua capacidade.

- Laboratório remoto

  - Esta configuração consiste num laboratório em que uma ou mais experiências podem ser realizadas sem intervenção direta, sendo comandada remotamente e os resultados obtidos por medições e/ou visualização de imagens em tempo real da experiência.

    Vantagens: possibilidade de funcionamento contínuo e acesso a partir de qualquer local, o que permite ao alunos obter resultados em qualquer altura e local [5, 51].

    Desvantagens: custo de montagem e a necessidade de dividir um único recurso por vários utilizadores o que leva a ser em muitas situações necessário agendamento.

2.3 Classificação dos sistemas encontrados em Automação Industrial

Os sistemas podem ser classificados quanto à finalidade, tangibilidade e ao controlo, podendo em cada uma destas classificações pertencer a uma ou mais classes.

2.3.1 Quanto à finalidade

- Sistemas profissionais

  - Desenvolvidos com a finalidade de utilização profissional como o meio industrial, estes sistemas visam a robustez e a fiabilidade de forma a garantir um funcionamento contínuo e sem falhas.
  - Muitos destes sistemas devido à sua estrutura podem ser potencialmente perigosos e deve ser garantido que no seu raio de utilização não circulem objetos estranhos nem pessoas, devendo existir sistemas de deteção de obstáculos que parem o sistema em caso de emergência.

- Sistemas didáticos
  - Tendo como finalidade a utilização em contexto educacional [60], estes sistemas são desenvolvidos de forma a replicar o funcionamento de sistemas industriais sem riscos para o utilizador e ambiente circundante.
  - Apesar da finalidade educacional estes sistemas são muitas vezes conectados a sistemas de controlo industrial [62], sendo o *software* e programação próxima do sistema real em que foi baseado.

2.3.2 Quanto à tangibilidade

- Sistemas físicos (ou reais)
  - Estes sistemas tem um meio palpável, sendo as entradas e saídas variáveis que interagem com o meio físico.
  - Permitem transportar e transformar matérias reais e são estes os sistemas que se podem encontrar numa linha de produção de uma industria.
- Sistemas virtuais (ou simulados)
  - Estes sistemas são intangíveis, sendo programas a executar num meio de processamento como um computador ou microcontrolador que simulam o comportamento de um sistema físico a partir de dados de entrada do utilizador.
  - Dependendo do grau de implementação do sistema virtual é possível ter como dados de entrada não só os parâmetros de entrada do sistema em si como também parâmetros de avarias e do ambiente que poderão alterar o comportamento das saídas.

2.3.3 Quanto ao controlo

- Sistemas discretos
  - A evolução do processo é definida por um conjunto de estados.
  - As transições entre estados são definidas por eventos.
  - O número de estados do sistema é finito.
- Sistemas contínuos
  - O estado do processo pode ser descrito através de uma variável analógica.
  - Existe entre a entrada e a saída do processo uma relação que pode ser descrita através de um modelo contínuo (como uma função de transferência).

2.4 Soluções existentes

Nesta secção estão descritas algumas soluções de sistemas existentes no mercado e, ao final, é possível ver a sua classificação dentro das divisões referidas anteriormente (ver tabela 2.1).

2.4.1 *Automation Studio*

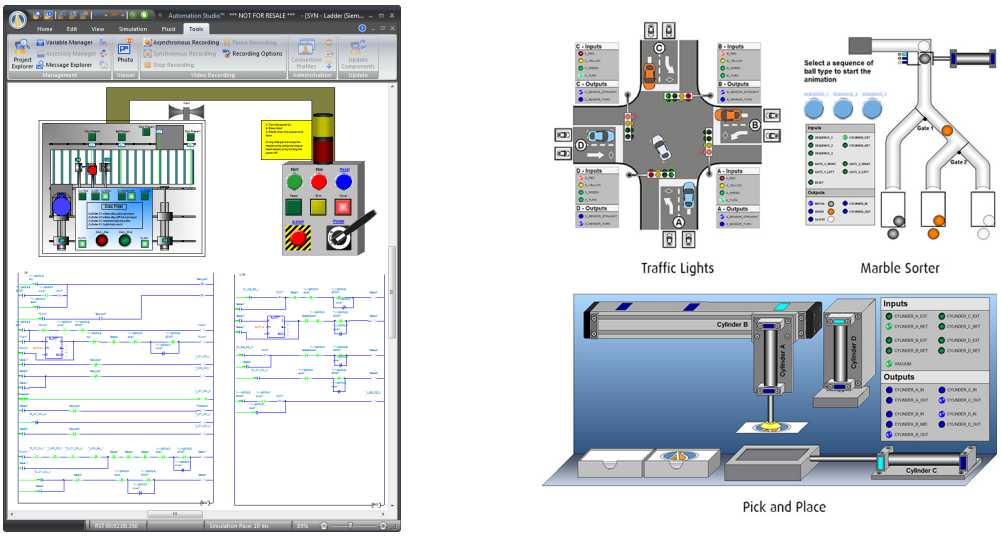
Desenvolvido pela *Famic Technologies Inc.* [15], este *software* permite o projeto e simulação de sistemas elétricos, hidráulicos e pneumáticos, sistemas de controlo, suportando bibliotecas de fabricante Allen Bradley e Siemens e a norma IEC 61131-3 [32]. Outras funcionalidades são o recurso a catálogos de fabricantes, dimensionamento de componentes, análise e resolução de falhas e comunicação por OPC e CAN Bus para ligação do *software* a *hardware* externo. Outras funcionalidades podem ser vistas na figura 2.1.

Figura 2.1: Diagrama com funcionalidades do *Automation Studio*

É fornecido em duas versões distintas, académica e profissional, com características adicionais pensadas na finalidade.

A versão académica disponibiliza uma plataforma de ensino onde é possível utilizar, modificar e criar recursos pensados no meio académico como guiões, aulas e exercícios interativos. O licenciamento desta versão também é pensado no meio académico permitindo a utilização remota tanto por professores como alunos.

A versão profissional por sua vez, idealizada para a industria, tem suporte para múltiplos utilizadores a trabalhar em simultâneo no mesmo projeto, criação de material para treino de funcionários e clientes, desenvolvimento de documentação técnica e partilha de projetos com clientes, fornecedores e colegas a partir de um gestor de direitos de acesso.



(a) Simulação de um programa em Ladder (b) Alguns dos sistemas virtuais disponíveis

Figura 2.2: Interface do *Automation Studio*

2.4.2 Festo Didactic

Secção da marca Festo [17] que fornece soluções dedicadas ao ensino, desde sistemas físicos a virtuais, tanto de processos contínuos como discretos. Algumas das soluções são descritas de seguida.

2.4.2.1 MPS

Abreviatura para *Modular Production System* (figura 2.3a), são sistemas físicos de controlo discreto que emulam um processo industrial montados em módulos que podem ser utilizados individualmente ou interligados de forma a emular uma linha de produção.

Os módulos são idealizados para serem ligados a sistemas de controlo como um Controlador Lógico Programável (*PLC*), sendo que o fabricante disponibiliza *software* que permite o controlo através do *EasyPort* (figura 2.3b), um dispositivo que permite conectar por USB um computador a um sistema didático físico do fabricante.

(a) MPS (b) *Easyport*

2.4.2.2 CIROS®

Este *software* (figura 2.4) permite criar modelos 3D de sistemas de automação e utilizar modelos já disponibilizados de sistemas físicos da Festo como os da gama MPS.

Permite simular em tempo real vários eventos como colisões no transporte, erros do sistema e características físicas de sensores. O controlo pode ser feito tanto por *softPLC* como *PLC*s por OPC ou eletricamente por *EasyPort*.

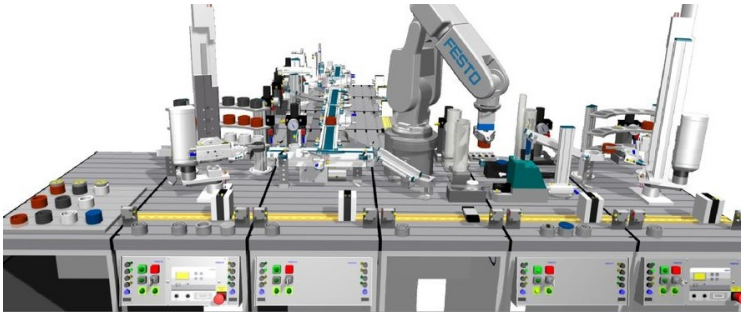


Figura 2.4: Sistema MPS simulado pelo CIROS

2.4.2.3 MecLab

Esta solução consiste num conjunto de sistemas físicos (figura 2.5) que replicam processos industriais reais e que, apesar da finalidade didática, são desenvolvidos na totalidade com componentes presentes na industria. Estes sistemas permitem ser alterados e expandidos e incluem vária documentação como a informação técnica, apresentações PowerPoint, exercícios e fichas de trabalho personalizáveis. Estas características levam a que esta solução seja uma opção pronta a usar para ensinar automação industrialtanto no ensino médio como superior.

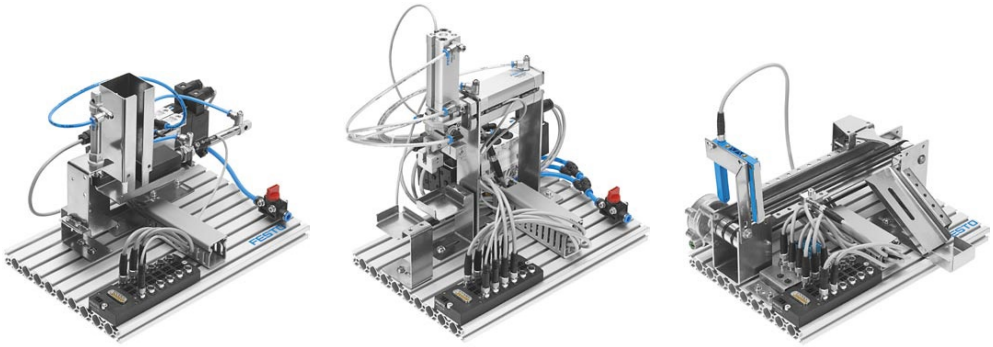


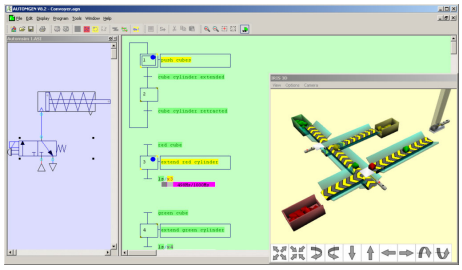
Figura 2.5: Sistemas MecLab

2.4.2.4 FluidSIM 5

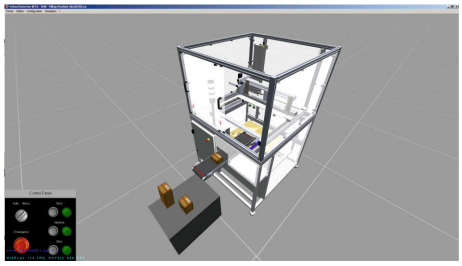
Este *software* disponibiliza a simulação de sistemas pneumáticos, hidráulicos e eletro-pneumáticos e permite o controlo a partir de **GRA**phe **F**onctionnel de **C**ommande **É**tape/**T**ransition (*GRAFCET*), podendo ser ligado a sistemas físicos pelo interface *EasyPort*.

2.4.3 *Automsim Premium* e *Virtual Universe Pro*

Publicitado pelo fabricante *IRAI* [37] como um concorrente económico do *Automation Studio* e *Fluidsim* possui igualmente a capacidade de desenvolvimento e simulação elétricos, pneumáticos, hidráulicos e de eletrónica digital e funções como a simulação de um *PLC* a serem garantidas pelo *softPLC Automgen*, outro *software* do mesmo fabricante. Na figura 2.6a é possível observar a utilização em conjunto destes *software* para realizar o controlo de um sistema virtual. Outro sistema virtual disponibilizado pela *IRAI* é o *Virtual Universe Pro* (figura 2.6b), um simulador 3D onde o utilizador pode utilizar óculos de realidade virtual para visualizar o mundo simulado e controlado diretamente por *PLC*s de várias marcas, *softPLCs* e simuladores de *PLC*, dispositivos compatíveis com o *Automgen*, *software* como o Matlab e o Labview e dispositivos que usem os protocolos Modbus TCP, SLMP ou *Open Platform Communications* (*OPC*).



(a) *Automsim Premium* a executar pelo *Automgen*



(b) Interface do *Virtual Universe Pro*

Figura 2.6: Interface dos *software* da *IRAI*

2.4.4 *ITS PLC* e *FACTORY I/O*

Estes *softwares* são simuladores em tempo real de sistemas industriais, desenvolvidos pela empresa *Real Games* [22] [23].

O *ITS PLC* possui cinco cenários de sistemas industriais discretos virtuais que conforme a versão do programa podem ser controlados por *PLC*s genéricos através de um módulo de interface (figura 2.7), *softPLC Automgen* ou por *WinSPS-S7* e *WinPLC-Engine*, ferramentas da Siemens de *softPLC* e programação de *PLC*.

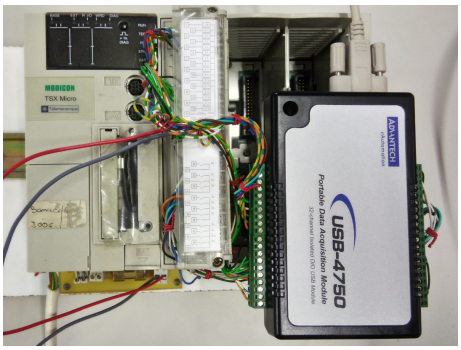


Figura 2.7: Simulação duma misturadora e módulo de interface do *ITS PLC* ligado a um autómato industrial

O *FACTORY I/O* é o sucessor do *ITS PLC*, vindo com mais de vinte cenários de sistemas industriais virtuais, permitindo ainda a criação de novos cenários. Outra funcionalidade adicionada é a introdução de variáveis analógicas deixando o controlo ser exclusivamente discreto. Além das opções já existentes no *ITS PLC*, foram introduzidas novas *drivers* de comunicação passando a permitir a ligação e controlo por *OPC Client Data Access* e *Modbus TCP/IP*.

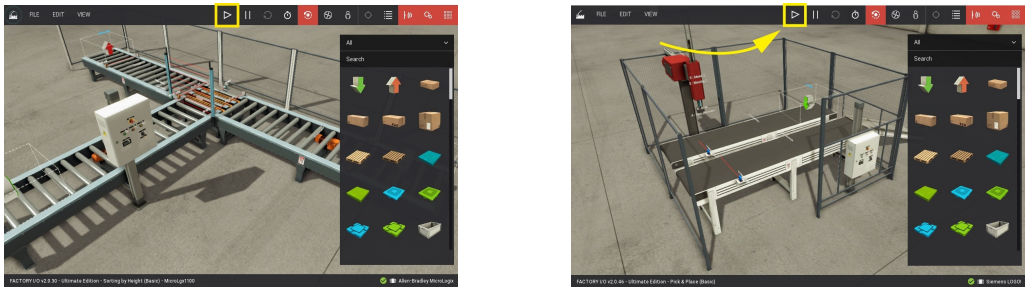
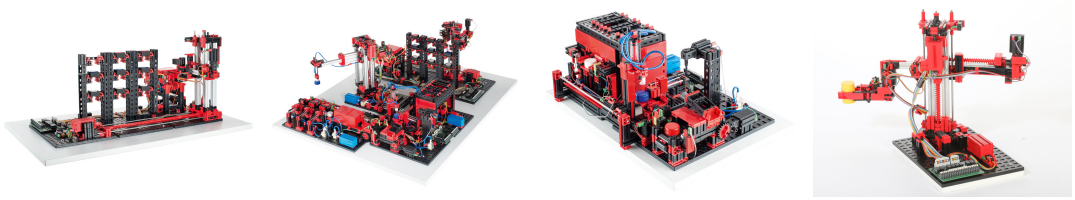


Figura 2.8: Exemplos de cenários do *FACTORY I/O*

2.4.5 *Fischertechnik Simulating*

Para além dos seus produtos destinados a um publico mais infantil com blocos de construção livre, a *Fischertechnik* também possui uma gama de produtos didáticos entre os quais se destaca a sua secção de simulação [28] com os modelos de treino, sistemas didáticos físicos desenvolvidos com recuso a blocos de construção da marca.

Os modelos são baseados em aplicações reais e equipados com atuadores e sensores compatíveis com sistemas de controlo industriais. Na figura 2.9 estão representados alguns dos modelos disponibilizados.



(a) Armazém (b) Fábrica (c) Forno (d) Robo 3D

Figura 2.9: Alguns dos sistemas didáticos *Fischertechnik*

2.4.6 Staudinger

Esta empresa é especializada em várias áreas de engenharia relacionadas com a industria, tendo uma secção dedicada à simulação [29] onde são desenvolvidos sistemas físicos didáticos construídos a partir de componentes *Fischertechnik* e industriais baseados em sistemas reais.

Os sistemas didáticos são disponibilizados em modelos *standard*, compactos e combinacionais, todos eles preparados para ligação a sistemas de controlo industriais como os *PLC*. Os modelos *standard* têm como objetivo a simulação das funções básicas de um sistema real de forma

individual. Os compactos diferem dos *standard* por terem uma dimensão inferior, facilitando o seu transporte, menor detalhe de funções e um custo inferior. Os modelos combinacionais por sua vez diferem dos anteriores por poderem ser utilizados de forma independente ou interligados.

Graças a modularidade destes sistemas o fabricante disponibiliza a possibilidade de desenvolver novas soluções adaptadas às necessidades do cliente.

(a) Linha de Processo Flexível com Portal de 3 Eixos

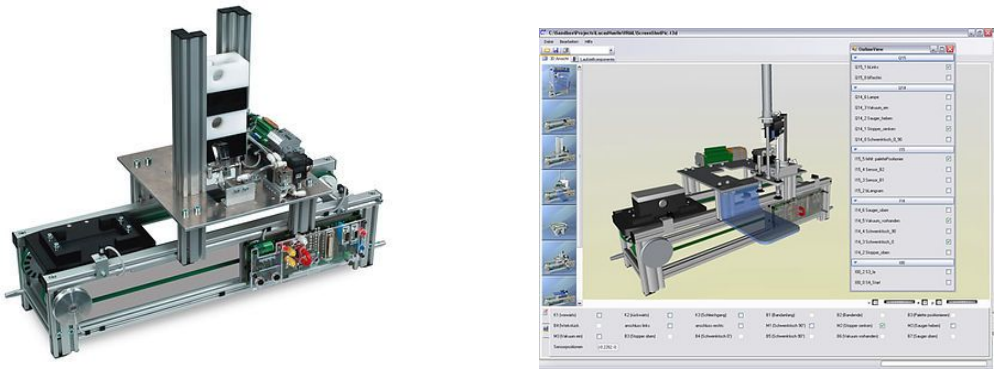
(b) Prensa excêntrica

(c) Quarto temperado

Figura 2.10: Alguns dos sistemas didáticos Staudinger

2.4.7 Lucas Nülle

Esta empresa desenvolve e fabrica sistemas didáticos para formação profissional inicial e contínua em várias áreas da engenharia eletrotécnica e mecânica como circuitos eletrónicos e elétricos, micro-controladores, automação, controlo, pneumática e hidráulica.Dentro destas áreas são produzidos sistemas didáticos físicos que são disponibilizados com todo o equipamento necessário para realizar as experiências, existindo em algumas áreas, como o caso de automação, simuladores dos sistemas físicos. É também disponibilizado *software* que possui em algumas das áreas cursos para auto aprendizagem e o controlo e obtenção de resultados das experiências [53].



(a) Sistema didático físico

(b) Sistema didático virtual que simula um sistema físico do fabricante

Figura 2.11: Produtos na área de automação industrialda Lucas Nülle

2.4.8 FEUPAutom e SimTwo

Este *software* consiste num *softPLC*, FEUPAutom, com ligação a conjunto de sistemas virtuais didáticos, SimTwo, e foi desenvolvido por docentes da FEUP para ser utilizado como uma experiência simulada em laboratórios locais com a possibilidade de os alunos poderem continuar o trabalho em casa. O FEUPAutom suporta linguagens de programação bastante próximas de ST e *GRAFCET* e comunicação *Modbus Transmission Control Protocol (TCP)/Internet Protocol (IP)* para permitir o controlo de sistemas que suportem o protocolo [67].

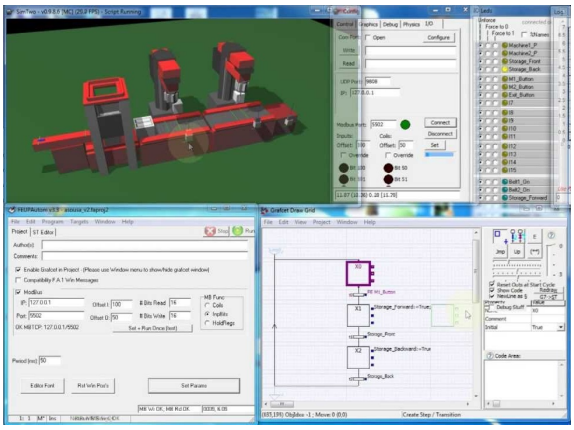


Figura 2.12: Interface do FEUPAutom e do SimTwo

Tabela 2.1: Classificação dos diferentes sistemas analisados

Fabricante	Produto	Finalidade		Meio		Controlo	
		Industrial	Didático	Físico	Virtual	Discreto	Contínuo
Famic Technologies Inc.	Automation Studio Professional	✓			✓	✓	✓
	Automation Studio Educational		✓		✓	✓	✓
Festo Didactic	MPS®		✓	✓		✓	✓
	MecLab®		✓	✓		✓	
	FluidSIM® 5	✓	✓		✓	✓	✓
	CIROS®	✓	✓		✓	✓	✓
IRAI	Automsim Premium	✓	✓		✓	✓	✓
	Virtual Universe Pro	✓	✓		✓	✓	✓
Real Games	ITS PLC		✓		✓	✓	
	FACTORY I/O		✓		✓	✓	✓
Fischertechnik Simulating	Soluções várias		✓	✓		✓	✓
Staudinger	Soluções várias		✓	✓		✓	✓
Lucas Nülle	Soluções várias		✓	✓	✓	✓	✓
FEUP	FEUPAutom e SimTwo		✓		✓	✓	

Capítulo 3

Sistema didático existente

Neste capítulo é descrito o sistema didático sobre o qual foi realizado este trabalho e as suas componentes física e lógica.

O sistema didático é um recurso de apoio no ensino de automação industrial, utilizado em laboratório por alunos e docentes. Este recurso permite desenvolver e testar soluções de processos de controlo discretos que podem ser encontradas em situações reais de uma linha de produção.

3.1 Especificações da componente física

O sistema didático é baseado numa solução personalizada da Staudinger composta por cinco células, armazém, máquinas série, máquinas paralelo, montagem e carga e descarga, que correspondem à componente física (figura 3.1).

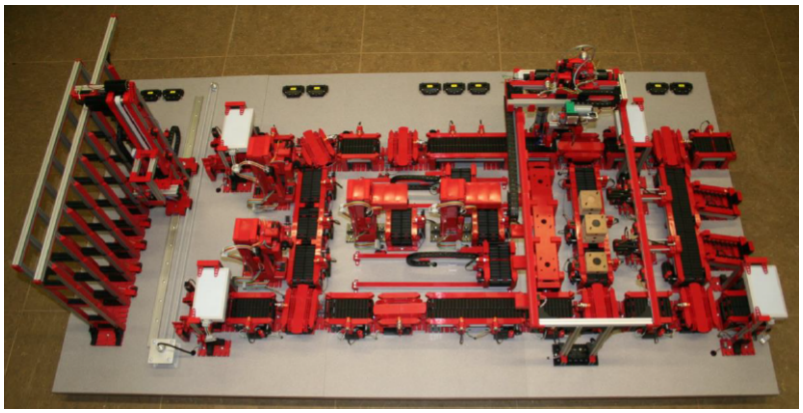
As células podem ser utilizadas de forma independente ou formando um *Flexible Manufacturing System* (*FMS*) quando interligadas. Estes sistemas têm como característica a capacidade de se adaptar a mudanças, previstas ou não, na manufatura [9]. Esta capacidade provém do facto de redundância nos caminhos de transporte e máquinas com suporte para as transformações necessárias.

De forma a transportar os materiais, o sistema didático é composto por vários tapetes, todos eles bidirecionais, de três tipos:

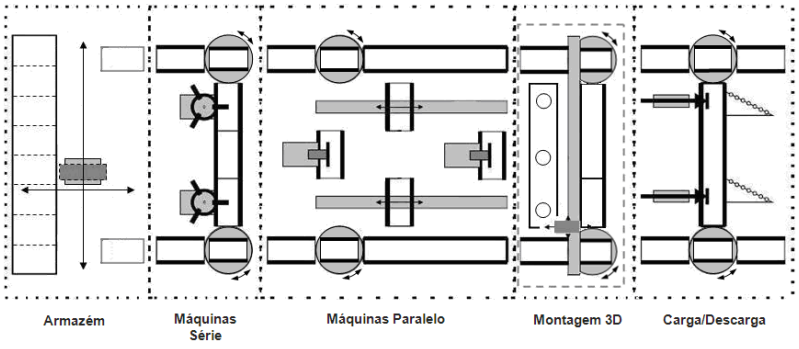
- Lineares - Apenas movem as peças num eixo.
- Rotativos - Podem rodar 90º e mudar o eixo de movimento das peças.
- Deslizantes - Podem se mover no eixo perpendicular ao movimento das peças.

Os materiais são simulados por peças de madeira com pontos de metal nas suas laterais para deteção pelos sensores indutivos presentes nos tapetes e uma *tag RFID* única para identificação.

As células são descritas em mais detalhe de seguida, estando representado para cada uma o seu esquema com a localização dos vários componentes. Os sensores são representados por pequenos retângulos a verde para contacto normalmente aberto e azuis para contacto normalmente fechado.



(a) Aparência



(b) Esquema

Figura 3.1: Estado inicial do sistema didático

3.1.1 Armazém

Esta célula é composta por locais de armazenamentoe permitem armazenar até 3 peças empilhadas entre si por local de armazenamento. De forma a movimentar as peças a célula possui um sistema de transporte capaz de colocar e retirar as peças dos locais de armazenamento e as transferir para os tapetes.

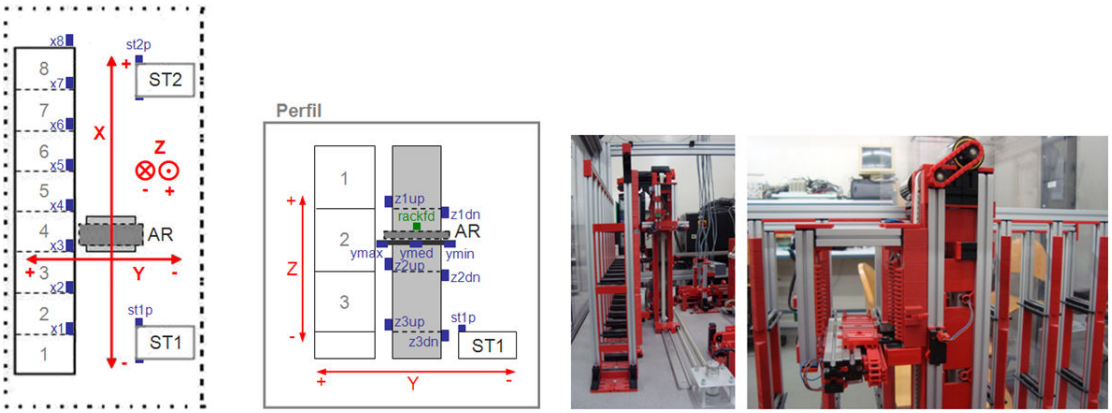


Figura 3.2: *Layout* e aspeto da célula armazém

3.1.2 Máquinas série

Esta célula é constituída por duas máquinas posicionadas em série, tendo uma peça de passar pela zona de maquinação de ambas as máquinas para atravessar a célula.

As máquinas têm como movimentos possíveis a deslocação horizontal para se aproximar e afastar da zona de maquinação e a seleção de uma de três ferramentas e sua rotação.

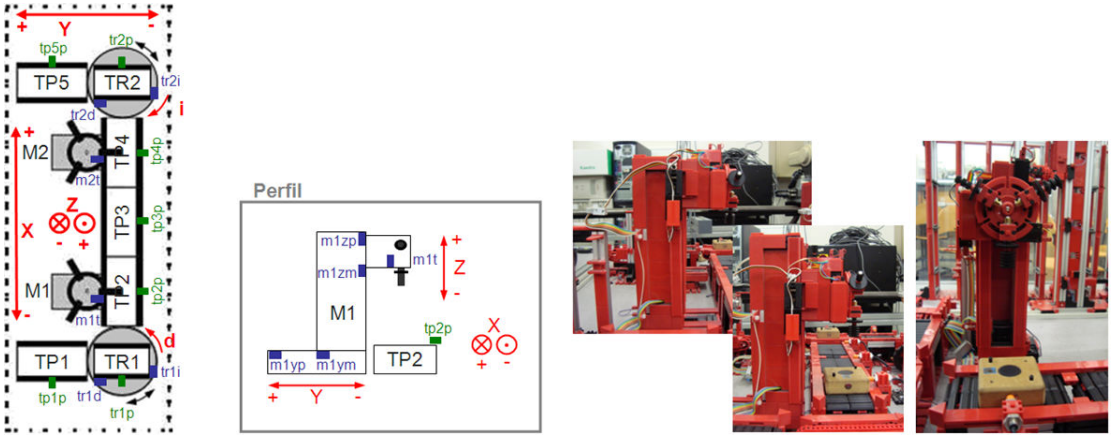


Figura 3.3: *Layout* e aspeto da célula de máquinas série

3.1.3 Máquinas paralelas

Esta célula é constituída por duas máquinas posicionadas em paralelo sendo que uma peça para atravessar a célula apenas necessita de passar pela zona de maquinação de uma das máquinas.

As máquinas desta célula têm como movimentos possíveis uma deslocação horizontal para se aproximar e afastar da zona de maquinação e um movimento vertical para aproximar a ferramenta da peça.

O transporte de peças entre as máquinas é feito por tapetes deslizantes.

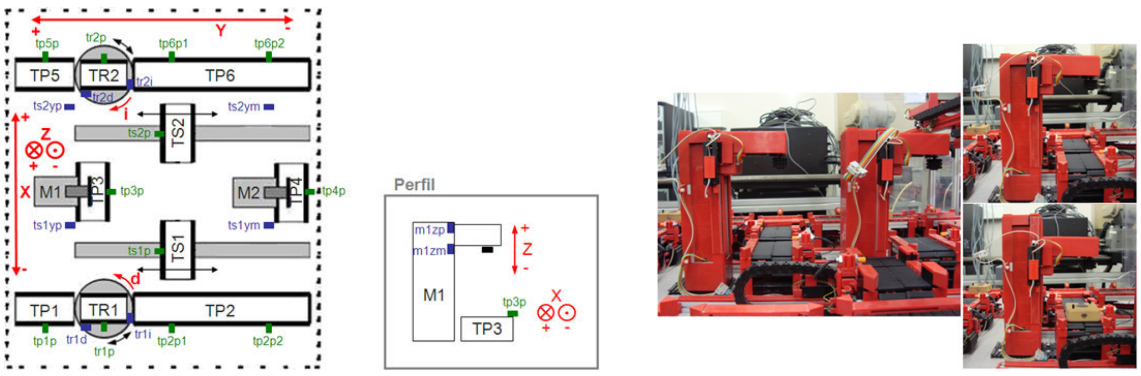


Figura 3.4: *Layout* e aspeto da célula de máquinas paralelo

3.1.4 Montagem

Esta célula é constituída por um sistema de transporte com uma pega atuada por pneumática capaz de se mover em três eixos capaz de aceder a todos os tapetes da célula e mesas de montagem que permitem empilhar peças.

A célula possui proteções elétricas, não representadas no *layout*, que desligam os motores para evitar que sejam ultrapassadas as posições limite de funcionamentos na célula nos eixos X e Y de forma a ser possível a inicialização segura do sistema quando a posição inicial é desconhecida.

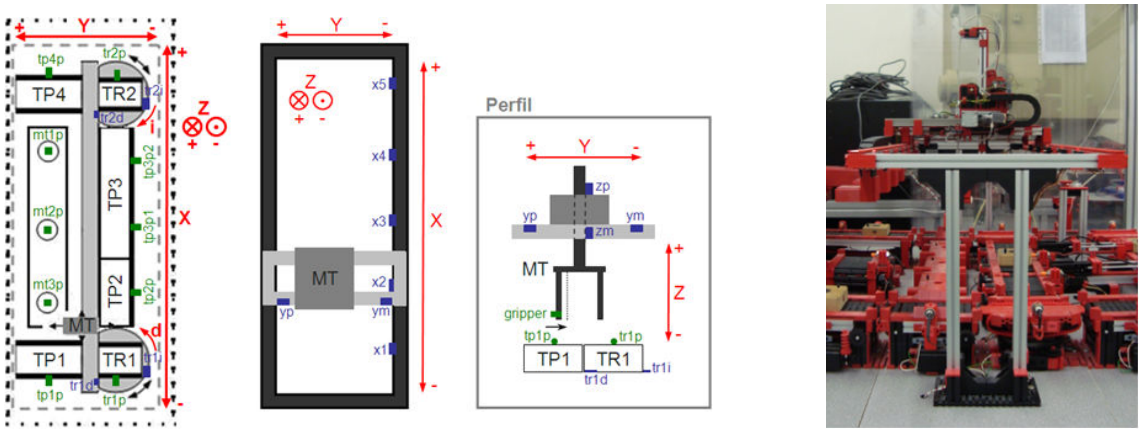


Figura 3.5: *Layout* e aspeto da célula de montagem

3.1.5 Carga e descarga

Esta célula possui dois pontos de descarga onde as peças são transferidas por *pushers* e dois tapetes rotativos que permitem colocar ou descarregar peças. Os pontos de descarga têm sensores para detetar se estão livres para receber uma peça, tendo capacidade para duas peças. A remoção das peças dos pontos de descarga é feita de forma manual.

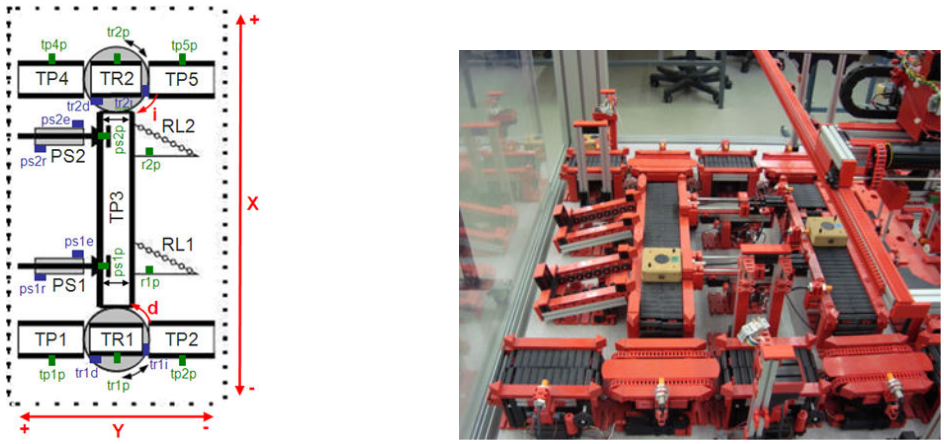
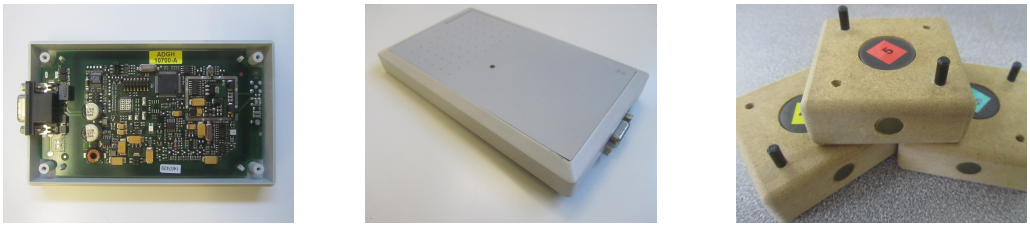


Figura 3.6: *Layout* e aspeto da célula de carga e descarga

3.1.6 Identificação de peças por *RFID*

As peças utilizadas possuem uma *tag RFID* com uma identificação única respeitando a norma *ISO15693* [42, 43, 44] com uma frequência de 13,56MHz. O sistema foi fornecido com quatro leitores *RFID* que permitem ler várias *tag* em simultâneo graças à função de anti-colisão e com um alcance de 18 centímetros [14]. Estes leitores possuem uma ligação DB9 capaz de comunicar localmente por RS-232 ou RS-485.

O aspeto do leitor e das peças com a *tag* podem ser observados na figura 3.7.



(a) Interior (b) Exterior (c) Peças utilizadas

Figura 3.7: Leitor *RFID* do sistema didático

3.1.7 Identificação de peças por código de barras

O sistema de identificação por *RFID* com que o sistema didático veio equipado tem o inconveniente de estar localizado apenas nas células Armazém e Carga e descarga. De forma a possibilitar identificar as peças em outras células foi acrescentado um leitor de código de barras e etiquetas com um identificador único nas peças de madeira.

O valor das leituras dos código de barras são transmitidos através de um registo *Modbus TCP/IP*.

Figura 3.8: Sistema de identificação por código de barras

3.2 Especificações da componente lógica

3.2.1 Base computacional e *Interlock*

O *Interlock* é um programa com a função principal de evitar a ocorrência de situações que possam danificar o sistema didático. De forma a ser possível executar o *Interlock* foi necessária a introdução de uma base computacional no sistema didático.

Os requisitos passam pela capacidade de processamento suficiente para execução dentro dos tempos de reação necessários para evitar danos, interface *Ethernet* RJ-45 de forma a ser possível a ligação à rede de comunicação e suporte para o protocolo *Modbus TCP/IP*. A escolha passou pela ICnova AP7000 Base [35], uma placa que executa uma versão personalizada de linux e que possui comunicações *Ethernet*, I^2C e *UART*, além de pinos de entrada e saída digitais.

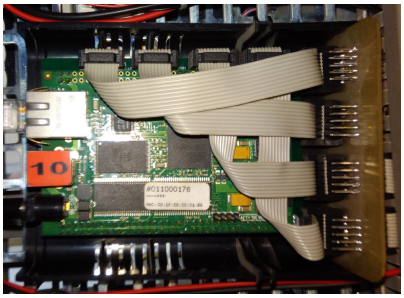


Figura 3.9: ICnova AP7000 instalada

De forma a conhecer os comandos do utilizador, o *Interlock* gere um servidor *Modbus* onde o sistema de controlo se conecta, onde são disponibilizados os valores dos sensores do sistema didático e é feito o mapeamento dos atuadores.

Os valores dos atuadores do servidor *Modbus* é verificado pelo *Interlock*, sendo bloqueadas atuações que para o estado atual do sistema didático, podem levar a situações de dano. Alguns exemplos dessas situações é a passagem dos limites físicos de componentes, comandos contraditórios para um atuador ou uma peça ficar presa e bloquear a rotação de um tapete rotativo.

A verificação é feita através de um conjunto de regras implementadas no *Interlock* e os valores dos sensores e ordens para os atuadores do sistema didático é feita através de de módulos de entradas e saídas remotas, apresentados na secção 3.2.2.

O *Interlock* também é responsável pelo controlo do semáforo colocado na plataforma de suporte do sistema didático. O significado das diferentes cores indicam os seguintes estados:

- Vermelho - Paragem de emergência ativa (secção 3.2.4).
- Laranja - *Interlock* com pelo menos um bloqueio ativo.
- Azul - *Interlock* com um bloqueio ativo nos últimos 15 segundos.
- Verde - Sistema inicializado corretamente.

Na figura 3.10 é possível ver a arquitetura do *Interlock* e como se interliga aos restantes sistemas.

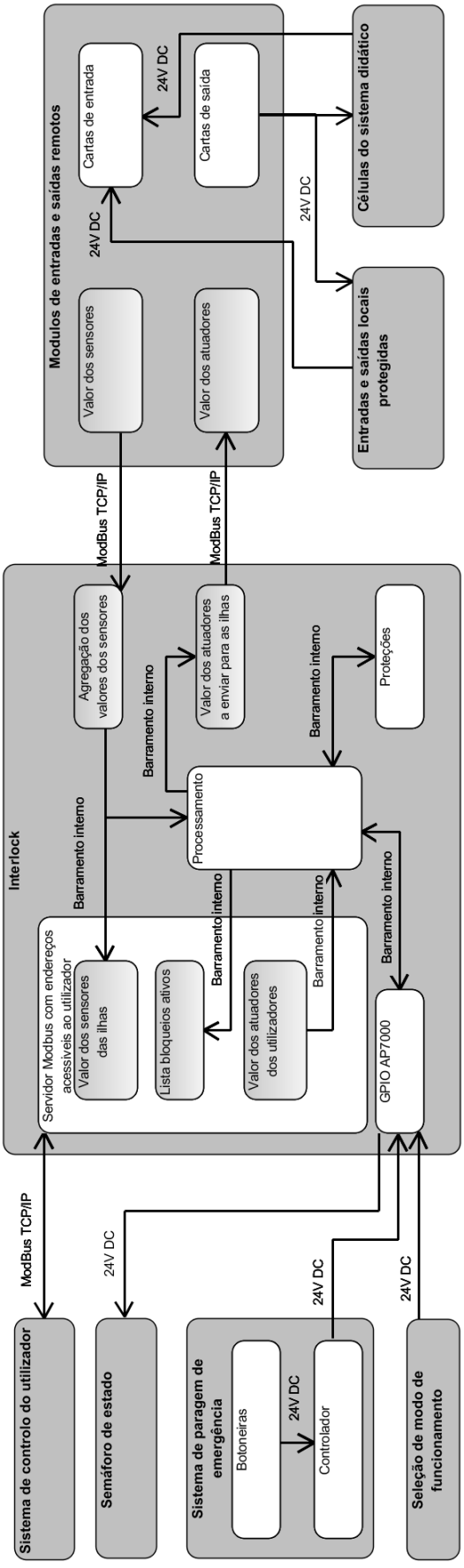


Figura 3.10: Esquema lógico de funcionamento do *Interlock* com o restante sistema didático

3.2.2 Módulos de entradas e saídas remotas

Para flexibilizar a utilização do sistema didático as entradas e saídas das várias células foram ligadas a módulos de entradas e saídas remotas da gama de produtos Modicon STB da Schneider Electric [10, 11, 12], que permitem a ligação de sensores e atuadores a um *PLC* através de vários tipos de rede como CANopen, DeviceNet, Profibus DP ou Ethernet. A utilização destes módulos permite evitar usar cabos elétricos longos entre os postos de trabalho e o sistema didático que são sujeitos a interferências e perdas elétricas.

Estes módulos são compostos por uma carta de comunicações que define o tipo de rede utilizado e uma ou mais fontes de alimentação, sendo expansíveis através de vários tipos de cartas. Estas cartas podem ser de entrada e saída, permitindo a ligação tanto de sensores e atuadores analógicos como digitais ou cartas de funções especiais como contadores, sendo possível desta forma montar um módulo à medida das necessidades.

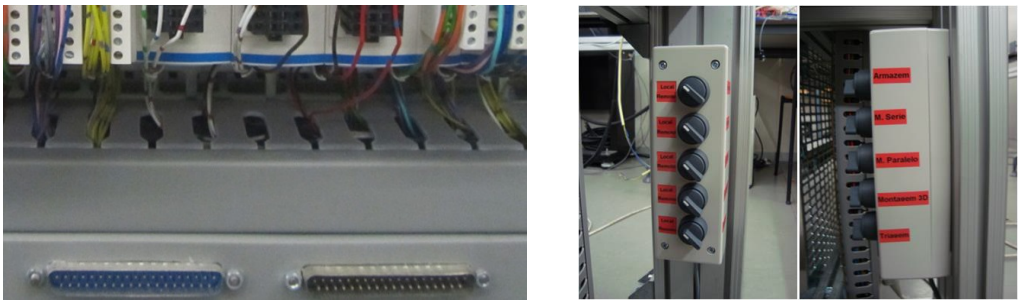
O sistema didático foi equipado com módulos com cartas de comunicação STB NIP2212 (figura 3.11) que permitem a interface das entradas e saídas das células em *MODBUS TCP/IP*. Este solução permite a comunicação e controlo através do *Interlock* ou outro sistema de controlo capaz de utilizar *Modbus TCP/IP* .

Figura 3.11: Módulos de entradas e saídas remotas instalados

3.2.3 Modo local com proteção

De forma a permitir o controlo do sistema didático, atuando diretamente através de sinais de 24VDC, foram adicionadas por baixo dos módulo de entradas e saídas remotas fichas *DB37* (figura 3.12a) que replicam as fichas das células. Estas entradas e saídas estão ligadas aos módulos de entradas e saídas remotas de forma a ser possível ao *Interlock* efetuar a verificação das proteções.

A seleção entre modo local ou remoto é feita por célula através de um conjunto de botões numa botoneira instalada na estrutura do sistema didático (figura 3.12b). A utilização deste modo exige que entre o sistema e a bancada de trabalho haja um fio por cada sinal a utilizar mais um fio para a referência (*GND*).



(a) Fichas DB37 para ligação do modo local (b) Botoneira para seleção de modo

Figura 3.12: Componentes do modo local

3.2.4 Sistema de paragem de emergência

De forma a ajudar a proteger o sistema de situações que o podem danificar, foram adicionadas quatro botoneiras de emergência, uma em cada extremidade do sistema didático, que permitem ao utilizador desativar todos os atuadores.

As botoneiras de emergência estão ligadas em série a um controlador de segurança da Schneider Electric (figura 3.13a), configurado para paragem de emergência com um canal e arranque automático, desligando a alimentação da componente de saída dos módulos de entradas e saídas remotas e dessa forma desativando o funcionamento de todos os atuadores do sistema didático.

De forma a evitar que o sistema possa ser utilizado de forma indevida, uma das botoneiras instaladas tem o desencrave através de chave (figura 3.13c). Isto permite utilizar esta botoneira como um interruptor do sistema, inibido o seu funcionamento até que alguém responsável faça o desencrave.

(a) Controlador emergência (b) Botoneira de emergência de desencrave mecânico (c) Botoneira de emergência com desencrave por chave

Figura 3.13: Componentes do sistema de paragem de emergência

3.2.5 Programa de demonstração

O sistema didático é um dos pontos de interesse nas visitas externas ao departamento pelo que este programa tem como finalidade mostrar todo o sistema didático em movimento. O ciclo do programa consiste em, de forma aleatória, retirar peças do armazém, transportar para um dos processos do sistema e depois as voltar a colocar no armazém. Situações de colocar ou retirar peças em tapetes parados está prevista, fazendo o ciclo do programa para essa peça arrancar ou parar desse ponto.

O programa é executado num *PLC* TSX Premium da Schneider Electric do laboratório, que por ser um recurso utilizado em aulas, sempre que se deseja fazer uma demonstração do sistema didático com este programa é necessário o programar, algo que exige algum conhecimento técnico de como proceder e por ser um pouco moroso necessita que a preparação deva ser realizada previamente.

3.2.6 Simulador do sistema didático

O sistema didático tem uma disponibilidade limitada, tanto pelo horário de funcionamento do laboratório onde se encontra instalado, como pelo facto de ser único e apenas permitir que um utilizador use o mesmo recurso do sistema didático. Caso mais de um utilizador tente comandar o mesmo recurso do sistema didático os comandos são sobrepostos levando a uma situação de liga/desliga nos atuadores.

De forma a minimizar essa limitação foi desenvolvido um simulador baseado em java do sistema didático (figura 3.14) que pode ser executado em vários sistemas computacionais. O interface de controlo do simulador é por *Modbus TCP/IP*, tendo o mesmo mapeamento que o sistema físico, permitindo que um mesmo programa possa controlar ambos os sistemas, bastando apenas alterar o endereço *IP* do sistema a controlar.

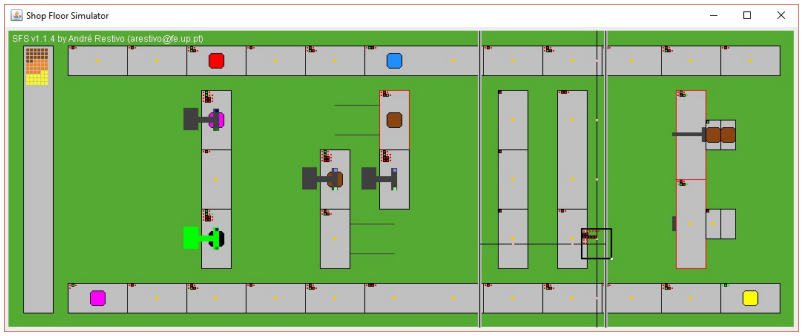


Figura 3.14: Simulador do sistema didático

O funcionamento do simulador é bastante próximo do físico mas situações reais como diferenças de velocidade entre tapetes e atrasos na passagem das peças entre tapetes não são tidas em conta, levando a que um programa perfeitamente funcional em simulação possa não o ser no sistema físico.

Situações indevidas, que seriam bloqueadas pelo *Interlock* no sistema didático, no simulador levam a uma avaria que é representada por uma moldura vermelha em volta do local onde ela ocorreu (observar os *pushers* da figura 3.14).

Capítulo 4

Soluções propostas

Neste capítulo são apresentadas as soluções propostas que esta dissertação visa implementar, divididas pela implementação ser ao nível lógico ou ao nível físico, antecedidas do levantamento de requisitos a partir das quais foram desenvolvidas.

4.1 Levantamento de requisitos

4.1.1 Limitações encontradas

De forma ser possível propor soluções para melhorar o sistema didático foram identificadas as suas limitações:

- Identificação das peças por *RFID* limitado a quatro pontos do sistema didático .
- A utilização do leitor de código de barras em algumas zonas do sistema didático implica o cabo passar por cima de tapetes e máquinas, podendo causar danos caso fique preso.
- Sistema de transporte da célula de montagem com dificuldade de funcionamento no eixo *Z*.
- Utilização de sistemas computacionais distintos para executar o *Interlock* e a identificação de peças por código de barras.
- O programa de demonstração necessita de um *setup* demorado e realizado por alguém com conhecimento técnico.
- A informação das proteções não é disponibilizada de uma forma que permita identificar facilmente quais estão ativas.
- Célula de montagem e célula de carga e descarga com funcionalidades limitadas para desenvolver trabalhos laboratoriais com um grau de complexidade semelhante às outras células.
- A falta de informação da ocupação do armazém não permite desenvolver proteções no *Interlock* para prevenir a colocação de peças em locais já ocupados.

4.1.2 Requisitos

Os requisitos propostos têm como base as limitações anteriormente descritas e imposições dos responsáveis e utilizadores do sistema didático:

- Compatibilidade com os sistemas de controlo já utilizados.
- Correção de limitações encontradas no sistema didático:
  - Funcionalidades adicionais para as células de montagem e de carga e descarga.
  - Informação de ocupação do armazém.
  - Expansão do sistema de identificação das peças.
  - Acesso simples à informação das proteções ativas.
- Programa de demonstração na base computacional.
- Minimizar alterações físicas que sejam irreversíveis.
- Melhor custo/beneficio.

4.2 Propostas a nível lógico

4.2.1 Alteração da base computacional

A base computacional onde é executado o *Interlock* inicial do sistema didático limita as funcionalidades que se desejam implementar, tendo já se tido a necessidade de recorrer a outra base computacional para fazer a interface do leitor de código de barras. De forma a ultrapassar estas limitações é proposta a alteração da base computacional para uma capaz de satisfazer os seguintes requisitos:

- Capacidade de processamento para executar as funcionalidades a implementar com margem para permitir a adição de outras funcionalidades no futuro.
- Arquitetura compatível com soluções de *softPLC* existentes para desenvolvimento de um novo *Interlock*.
- Suportar rede Ethernet para permitir a conexão aos módulos de entradas e saídas remotas.
- Portas USB com capacidade de *Host* para ligação de periféricos.
- Suportar vários protocolos de comunicação como I^2C, SPI e UART, nativo ou por extensão.
- Baixo preço.

4.2.2 *Interlock* compatível com as novas especificações

Visto o *Interlock* inicial ter sido desenvolvido para a base computacional onde é executado, a alteração da base computacional implica a necessidade de um novo *Interlock*.

O desenvolvimento de um *Interlock* sem recurso a um *software* específico é possível, criando uma aplicação de raiz, essa opção teria um grau de dificuldade elevado, requerendo a integração de várias bibliotecas para comunicação com os periféricos, garantindo a sua compatibilidade e criar formas de conseguir realizar a depuração de problemas de funcionamento.

A utilização de um *softPLC* fornece uma base de desenvolvimento em que a comunicação com os vários periféricos já está validada, assim como a possibilidade de depuração de problemas de funcionamento do código desenvolvido.

É por isso proposta a solução de desenvolver um *Interlock* baseado num *softPLC* compatível com a base computacional selecionada. De forma a permitir conectar e integrar as funcionalidades já existentes assim como as funcionalidades a adicionar pelas restantes soluções propostas, é necessário cumprir os requisitos:

- Escravo *Modbus TCP/IP* - Garantindo a compatibilidade com as características do *Interlock* original.
- Comunicação com os módulos de entradas e saídas remotas - Suportar *Modbus TCP/IP* e as funções *"Read Holding Registers"*, *Function Code* (*FC*)03, para aceder às entradas e *"Write Multiple Holding Registers"*, *FC*16, para atuar nas saídas dos módulos.
- Suporte para os leitores *RFID* - Permitir a comunicação com os leitores por um dos meios de comunicação (I^2C, SPI ou UART) suportados por ambas as partes.
- Suporte para o leitor de código de barras - Os leitores podem funcionar como um dispositivo *Human Interface Device* (*HID*), emulando um teclado, e/ou como uma porta série.
- Suporte *Human Machine Interface* (*HMI*) por ligação de um sistema *Supervisory Control And Data Acquisition* (*SCADA*) ou capacidade de gerir painéis de informação localmente.
- Capacidade de controlar os dispositivos locais como o semáforo.

4.2.3 Expansão da identificação das peças por *RFID*

É proposto a instalação de leitores *RFID* em todas as células nos tapetes lineares por onde as peças são recebidas e enviadas para outras células sem que as condições de utilização dos tapetes sejam influenciadas.

O estudo a efetuar para selecionar a opção a implementar deve ter em conta a dimensão e alcance do leitor de forma a ser instalado por baixo dos tapetes e ser capaz de ler as *tag* das peças transportadas e o preço total da solução, não sendo restrição a utilização das *tag* já existentes.

4.2.4 Modificação do sistema de identificação por código de barras

Na instalação inicial para serem feitas leituras de código de barras em alguns tapetes implica o cabo do leitor passar por cima dos vários componentes do sistema didático, havendo o risco de causar estragos. Para evitar essa situação é proposto estudar qual a melhor opção entre o encaminhamento do cabo na estrutura do sistema didático e instalar um suporte que faça a recolha do cabo ou substituir o leitor por um modelo sem fios, fazendo a implementação da opção escolhida.

4.2.5 Painéis de informação e configuração do sistema didático

De forma a ser possível saber o estado do sistema didático de forma fácil e direta é proposta a criação de um painel, local ou acessível remotamente, que identifique as proteções ativas, o estado dos atuadores e sensores, comunicação com os módulos de entradas e saídas remotas e permita configurar as opções do sistema didático.

As opções possíveis passam por recorrer a uma consola industrial existente no laboratório, que permitem a comunicação com o sistema didático por *Modbus TCP/IP*, ou um sistema de *SCADA* a executar diretamente na base computacional ou acessível remotamente por página *WEB*.

4.2.6 Programa de demonstração inserido na base computacional

Para demonstrar o funcionamento do sistema didático a visitantes sem a necessidade de uma preparação prévia do *setup*, é proposto desenvolver um novo programa de demonstração que seja executado na base computacional. A solução a desenvolver deve apenas ser ativada por utilizadores autorizados e o seu funcionamento prever que peças possam ser retiradas ou colocadas durante a demonstração pelos visitantes, prevenindo bloqueios na execução do programa.

4.3 Propostas a nível físico

4.3.1 Correção da limitação mecânica da célula de montagem

De forma a remover a dificuldade do sistema de transporte da célula se mover no eixo Z é proposto estudar as seguintes soluções:

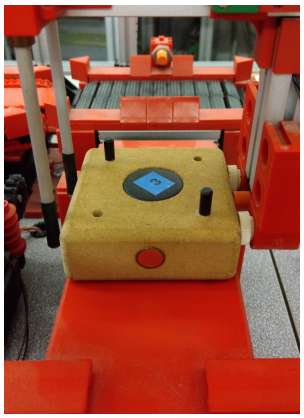
- Instalação de um contra-peso.
- Redução de peso do sistema de transporte.
- Redução de atritos.
- Substituição do motor.

E implementar as soluções que depois do estudo sejam validadas. No final da implementação desta proposta é esperado que a célula de montagem possa ser utilizada sem restrições.

4.3.2 Máquinas de simulação de processos para a célula de montagem

Esta célula possui uma mesa de montagem (figura 4.1a) com três posições que na configuração atual apenas são utilizadas para armazenar peças, sendo proposto adicionar em duas destas posições máquinas que simulem processos de transformação sem que o funcionamento atual da célula seja afetado. Para tal, visto o espaço necessário para colocar as peças e funcionamento do sistema de transporte (figura 4.1b) estas máquinas não poderão ultrapassar a altura da mesa.

(a) Posição livre



(b) Posição ocupada com sistema de transporte a carregar uma peça

Figura 4.1: Mesa de montagem da célula de montagem

4.3.3 Informação do estado de ocupação do Armazém

É proposta estudar formas de permitir verificar o estado de ocupação dos locais de armazenamento, sem interferir com o normal funcionamento. Para o mecanismo de carga e descarga da célula funcionar normalmente a instalação das opções selecionadas não pode exceder o espaço entre locais de armazenamento, 1,5 centímetros, nem posicionadas no seu interior. A base computacional e os módulos de entradas e saídas remotas têm limite de entradas disponíveis inferior aos 24 locais de armazenamento que se pretende monitorizar, pelo que se propõe também o estudo de formas de transmitir a informação da opção selecionada para deteção das peças.

Esta solução prevê que as opções selecionadas para deteção de peças e transmissão da informação para o *Interlock* sejam implementadas e o seu funcionamento validado.

4.3.4 Desenvolvimento de célula baseada num braço robótico

Esta proposta tem como objetivo criar uma nova célula baseada num braço robótico cujo *layout* provisório é representado na figura 4.2.

Composta por dois tapetes de transporte para ligação com as restantes células, três zonas para transformação de peças e o braço robótico para transporte das peças dentro da célula como recurso partilhado, tendo que cumprir os seguintes requisitos:

- Raio de ação - Capaz de aceder aos vários pontos que compõem a célula sem exceder os limites da mesma. O raio de ação está compreendido entre os 24 e 34 centímetros.
- Precisão - Movimentar entre as posições desejadas dentro de uma margem de tolerância que garanta que as peças são colocadas nas posições desejadas. Esta capacidade é dependente do tipo de motor e controlo utilizado.
- Eixos de movimento - Ter 4 eixos de liberdade ou 3 se a garra se mantiver horizontal de forma a efetuar as tarefas.
- Garra - Pegar nas peças do sistema didático pelas laterais de forma firme para que não exista deslizamento.
- Interface de controlo - Compatibilidade com a base computacional onde é executado o *Interlock*.

A nova célula substituirá no sistema didático a célula de carga e descarga pelo que terá de respeitar suas as dimensões.

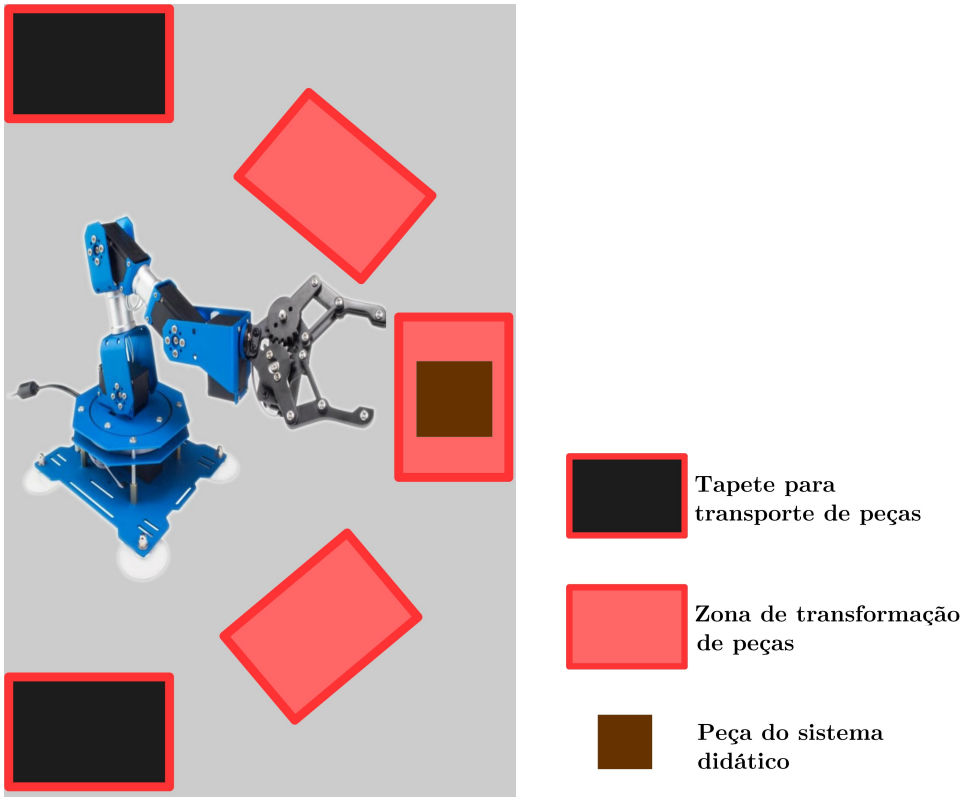


Figura 4.2: *Layout* provisório da nova célula.

Capítulo 5

Implementação e resultados

5.1 Alteração da base computacional

5.1.1 Estudo de opções disponíveis

As soluções iniciais para cumprir os requisitos desta proposta foram os *Embedded Single Board Computers* (*SBC*) e computadores pessoais nas vertentes micro PC, portátil e portátil híbrido. Como os *SBC* cumprem os requisitos com um custo inferior, entre 10 a 3 vezes face às restantes opções, apenas esta opção foi analisada em detalhe.

Foram analisadas algumas soluções dos fabricantes *BeagleBoard* [3], *Raspberry PI* [19] e [71] que cumprissem todos os requisitos, estando as suas características condensadas na tabela 5.1.

Tabela 5.1: Características dos *SBC* analisados

Fabricante	BeagleBoard		Raspberry Pi	UDOO		
Modelo	Green	Black	Model B	NEO Basic	NEO Full	X86 Basic
Processador	AM3358, 1GHz	AM3358, 1GHz	ARMv8-A 1.2 GHz	NXP™ i.MX 6SoloX com ARM Cortex-A9 e M4 embebidos		Intel Atom X5
Memória	1GB	512MB	1GB	512MB	1GB	2GB
Armazenamento	4GB	4GB + uSD	uSD	uSD	uSD	32GB eMMC + Sata
Ethernet	✓	✓	✓	✓	✓	✓
WiFi			✓		✓	
Bluetooth			✓		✓	✓
USB Host	✓	✓	✓	✓	✓	✓
UART	✓	✓	✓	✓	✓	✓
I^2C	✓	✓	✓	✓	✓	
CAN				✓	✓	
SPI			✓	✓	✓	
HDMI		✓	✓	✓	✓	✓
GPIO	✓	✓	✓	✓	✓	✓
Preço	37€	56€	43€	45€	58€	115€

Os *SBC* da gama NEO do fabricante *UDOO* possuem suporte nativo de *CAN*, um protocolo de comunicação utilizado em vários equipamentos de automação industrial, o que é uma mais valia para implementações futuras, contudo não é um requisito para as necessidades atuais. A gama *x86* da *UDOO* também é uma opção com características interessantes para o seu custo, contudo há soluções capazes de cumprir os requisitos a um custo inferior.

A *BeagleBoard* suporta algumas soluções de *softPLC* conhecidas [25] ao contrário dos NEO da *UDOO* em que todas as soluções de *softPLC* encontradas eram desconhecidas e de fontes não atualizadas.

A escolha para a base computacional foi a *Raspberry PI*. As suas características cumprem os requisitos necessários e existe um *know-how* tanto de utilização desta base computacional como das soluções de *softPLC* que suporta [26, 58].

5.1.2 Implementação

Inicialmente foi instalada a versão *standard* do *Raspbian*, o Sistema Operativo (S.O.) oficial para a *Raspberry PI*. Depois da escolha do *softPLC* numa fase posterior da dissertação, verificou-se que não ser necessário um ambiente gráfico na base computacional, sendo por isso substituído o S.O. para uma versão *light* sem ambiente gráfico do *Raspbian* de forma a maximizar os recursos disponíveis para as restantes aplicações.

A consulta às estatísticas do sistema operativo com o *softPLC* a executar todas as tarefas da última versão desenvolvida mostraram uma ocupação de processador inferior a 45% e de memória abaixo dos 3%.

5.1.3 Resultados obtidos

A *Raspberry PI* mostrou ser capaz de executar o *Interlock* selecionado na secção 5.2 sem saturar os seus recursos, tendo disponível mais de 50% do processador para outras tarefas que poderão ser adicionadas no futuro.

Um resultado negativo foi a limitação do número de dispositivos USB a conectar, detalhada na secção 5.3, que não é encontrada em nenhuma folha de característica do produto.

5.2 *Interlock* compatível com as novas especificações

5.2.1 Estudo de opções disponíveis

5.2.1.1 *openPLC*

O *openPLC* [57] é um *softPLC* também *opensource*, criado de acordo com o *standard* IEC 61131-3, que pode ser executado em várias plataformas como computadores com *Windows* ou Linux e dispositivos como *Raspberry PI* ou Arduino. Utiliza como ambiente de desenvolvimento o *PLCOpen*, que permite compilar os programas desenvolvidos para serem transferidos para a plataforma que os vai executar.

Os testes para comunicação com um módulo de entradas e saídas remotas falharam, não sendo possível obter os valores das entradas. O contacto com o desenvolvedor pelo fórum do produto [59] permitiu concluir que o *openPLC* não suportava a função *"Read Holding Registers"*, *FC*03, necessária para comunicar com os módulos de entradas e saídas remotas.

Outra limitação, que apesar de não essencial para o funcionamento também levaria a não utilizar esta opção, é não possuir suporte para depuração. Tal dificultaria validar o funcionamento do código desenvolvido e encontrar a origem de erros.

5.2.1.2 *Beremiz*

O *Beremiz* [4], também *opensource*, permite a compilação dos programas como executáveis para outras plataformas a partir do ambiente de desenvolvimento, o *PLCOpen*. Contudo na versão disponível à altura do projeto, opção de compilação e execução para a *Raspberry PI* a partir de uma outra máquina não estava disponível. De forma a contornar essa limitação, o *Beremiz* foi instalado diretamente na *Raspberry PI*, sendo para tal necessário que a versão de S.O. tenha ambiente gráfico para execução do ambiente de desenvolvimento.

Este *softPLC* possui capacidade de depuração e permite a ligação aos módulos de entradas e saídas remotas . Contudo não tem de forma nativa a capacidade de manipular ao bit os registos. Como as entradas e saídas de cada carta dos módulos são representados como um único registo de 16 bits, seria necessário desenvolver uma forma de manipular os registos ao bit para trabalhar as entradas e saídas individualmente. Esta manipulação levaria a atrasos no processamento e, no caso de erros no desenvolvimento das funções para manipulação, problemas na execução que poderiam ser difíceis de depurar.

Outra limitação encontrada no *Beremiz* foi a falta de um elemento para suporte da comunicação por porta série e outros meios de comunicação suportados pela *Raspberry PI*. Sem esta funcionalidade não é possível ligar diretamente os leitores que se pretende adicionar ao sistema didático ao *softPLC*.

A limitação do suporte de comunicações pode ser superada com o desenvolvimento de uma aplicação a executar diretamente na base computacional que fizesse a comunicação com os periféricos e a conversão dos dados obtidos para registos *Modbus TCP/IP* que seriam acessíveis ao *Beremiz* por rede.

5.2.1.3 *Codesys*

O *Codesys* [27] difere das opções anteriores, que são *open source*, por ser um *software* proprietário. A utilização do ambiente de desenvolvimento e *softPLC* em *Windows* é gratuita e sem limitações, contudo o funcionamento do *runtime* na *Raspberry PI* está limitado, na versão gratuita, a duas horas, tendo de ser reiniciado após desse período. O preço da licença para remover a limitação é de 50€, havendo a possibilidade da sua aquisição depois da validação do projeto.

O *Codesys* satisfaz todos os requisitos iniciais relacionados com o *Modbus TCP/IP* de forma nativa, inclusive a manipulação ao bit dos registos. Os requisitos relacionados com a ligação aos

leitores por porta série ou outros meios de comunicação também é satisfeita, havendo a possibilidade de implementar a comunicação série como terminal com a troca de mensagens de texto ou recorrendo ao protocolo *Modbus RTU*, assim como suporte para outros meios de comunicação suportados pela *Raspberry PI*.

O suporte para *HMI* também é satisfeito. É possível a ligação de um sistema *SCADA* e o desenvolvimento de painéis, acessíveis localmente e remotamente por páginas *web* com suporte HTML5. O servidor *web* para acesso aos painéis é parte integrante do *runtime* do *Codesys*.

O mapeamento e utilização do GPIO da *Raspberry PI* é suportado, sendo assim possível controlar dispositivos locais como o semáforo do sistema didático, sendo satisfeitos na totalidade os requisitos iniciais desta proposta.

5.2.2 Implementação

5.2.2.1 Definição da arquitetura da nova implementação

A arquitetura desta implementação está representada na figura 5.1.

Na nova arquitetura é também possível observar as especificações introduzidas no *Interlock* e novas especificações externas ao *Interlock* mas que são geridas por ele como o sistema de identificação de peças e a ocupação do armazém.

5.2.2.2 Ligação aos módulos de entradas e saídas remotas

De forma a implementar a arquitetura é necessário que o *softPLC* se conecte ao sistema didático através dos módulos de entradas e saídas remotas. Existem no total quatro módulos, um para o conjunto das células armazém e máquinas série e um por cada célula restante.

Os parâmetros a configurar para permitir a comunicação *Modbus TCP/IP* são

- Mestre - Os módulos são escravos *Modbus* pelo que é necessário configurar no *softPLC* um dispositivo *Modbus TCP/IP* mestre.
- Escravo - Para cada módulo é necessário adicionar a configuração de uma dispositivo *Modbus TCP/IP* escravo.
  - Endereço do módulo - Endereço *IP* e porta para comunicação *TCP/IP*.
  - Canal de comunicação - Por cada *FC Modbus* a utilizar é necessário configurar um canal de comunicação.
    - *FC* - Identificação do *FC* a utilizar.
    - Offset - Localização do primeiro registo a aceder.
    - Quantidade - Quantidade de registos a aceder.
    - Disparo - Momento em que a comunicação ocorre, pode ser cíclico ou atribuído a um evento.
    - Tratamento de erros - Comportamento a tomar se houver perda de comunicação, apenas em *FC* de leitura.

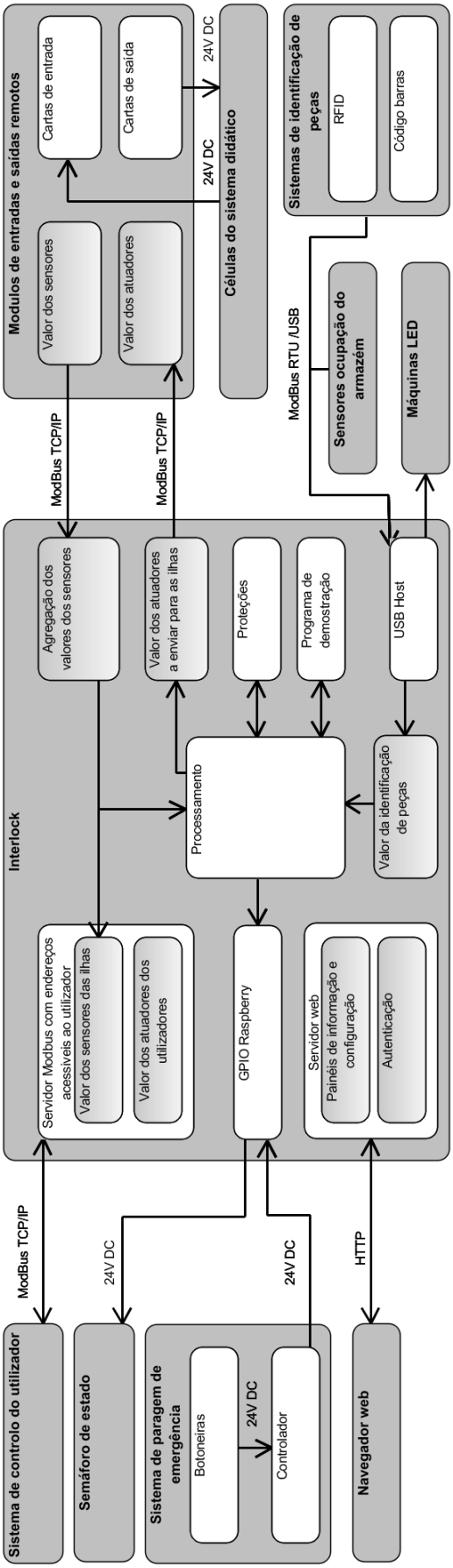


Figura 5.1: Esquema lógico de funcionamento do *Interlock* a implementar

Os parâmetros *Modbus* configurados estão representados na tabela 5.2, tendo sido feito e validado o mapeamento de todas as entradas e saídas do sistema didático antes de prosseguir para as restantes fases de desenvolvimento.

Tabela 5.2: Configurações dos módulos de entradas e saídas remotas

Célula	Configurações					
	Canal	FC	Offset	Quantidade	Disparo	Tratamento de erro
Armazém + Máquinas séerie	0	3	5391	23	Ciclo 10ms	Manter último valor
	1	16	0	14		-
Máquinas paralelo	0	3	5391	42		Manter último valor
	1	16	0	8		-
Montagem	0	3	5391	29		Manter último valor
	1	16	0	19		-
Carga / descarga	0	3	5391	18		Manter último valor
	1	16	0	6		-

5.2.2.3 Escravo *Modbus TCP/IP*

De forma a ser garantida a continuidade de controlo do sistema didático de forma remota é necessário que o *softPLC* seja acedido como um escravo (ou servidor) de *Modbus TCP/IP*.

Testes feitos ao funcionamento do escravo com recurso à ferramenta de simulação de um mestre *Modbus Modbus Poll* [65] demonstraram o bom funcionamento do escravo, podendo ser lido e escrito tanto com recurso a funções para variáveis discretas (*coils*) como registos (*words*).

Novos testes utilizando um *PLC* como mestre *Modbus* não foram realizados com sucesso. Isso foi devido ao facto de o *PLC* utilizado, um *TSX Premium* da *Schneider Electric*, utilizar inicialmente a função *"Read/Write Multiple registers"*, *FC*23. No caso de o dispositivo escravo não suportar a *FC*23, passa a utilizar as funções *"Read Holding Registers"*, *FC* 03 e *"Write Multiple Holding Registers"*, *FC*16 [13].

O *Codesys* suporta a *FC*23 mas para funcionar corretamente é necessária a ativação do modo *Holding - and - Input Register Data Areas overlay* nas configurações do escravo (figura 5.2). Após este ajuste nas configurações tanto a leitura como escrita de registos no escravo a partir do *PLC* passaram a ser realizadas com sucesso.

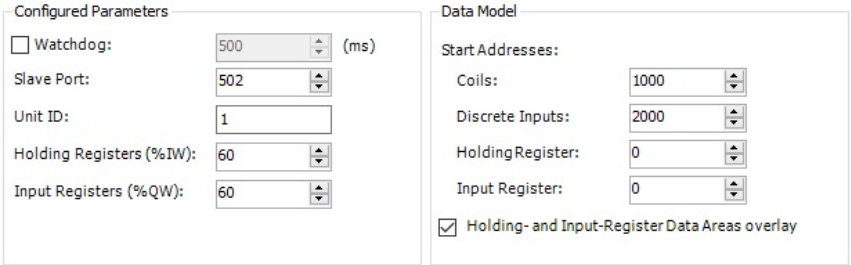


Figura 5.2: Configuração do escravo *Modbus TCP/IP* no *Codesys*

5.2.2.4 Comunicação por porta série

O *Codesys* não permite o funcionamento *"out of the box"* de dispositivos porta série na *Raspberry PI*, sendo necessário efetuar algumas configurações.

A instalação do *RunTime* do *Codesys* na base computacional cria um ficheiro de configurações na raiz do S.O. da *Raspberry PI*, sendo a configuração necessária é possível de ser consultar no fórum do produto [24].

No ficheiro de configuração deve ser adicionado um campo *"SysCom* e as suas configurações:

```
[SysCom]
Linux.Devicefile=/dev/PortTypeName
portnum := COM.SysCom.SYS_COMPORT1;
```

"PortTypeName" é o descritivo do tipo de porta a mapear, não sendo suportada a utilização de mais do que um descritivo.

Esta limitação foi superada com recurso a um *symlink* dos diferentes tipos de portas série para um novo tipo, *ttySerial*, realizado no arranque da *Raspberry PI*. Desta forma podem ser utilizados Arduino originais, descritos como *ttyACM*, como clones Arduino e outros periféricos USB que emulem uma porta série, descritos como *ttyUSB*.

Um extrato do código que realiza o *symlink* está abaixo representado:

```
echo "Mapping Serial Ports..."
ln -s /dev/ttyACM0 /dev/ttySerial0
(...)
ln -s /dev/ttyUSB12 /dev/ttySerial13
echo "Done!"
exit 0
```

A validação do funcionamento da comunicação, utilizando um exemplo disponibilizado pelo fabricante para a troca de mensagens de texto, mostrou ser funcional mas complexo. Exigia desenvolver o código para a configuração de todos os parâmetros da comunicação assim como para o controlo da porta e armazenamento da informação transmitida.

Uma alternativa nativa que permite simplificar o processo é a utilização da porta série usando o protocolo *Modbus RTU*. Esta solução permite configurar facilmente toda a comunicação entre o *Codesys* e o periférico e definir onde guardar a informação transmitida.

A desvantagem desta solução é a necessidade de os periféricos a conectar suportarem o protocolo. Visto existirem várias bibliotecas de *Modbus RTU* para Arduino e o baixo custo do microcontrolador é possível desenvolver um conversor de protocolo entre os periféricos e a base computacional, tornando a opção viável. Na secção 5.3 é feita uma descrição em pormenor de como o protocolo é implementado e configurado.

5.2.2.5 Levantamento das proteções ao sistema didático

Antes da fase de desenvolvimento foi feito um levantamento de todas as proteções a criar. Na listagem são apresentadas as condições em que as proteções são ativadas, divididas pelos diferentes componentes do sistema didático.

- Tapete linear
  - Ativação do tapete para ambos os sentidos em simultâneo.
  - Ativação do tapete para transferência de peça se o tapete tiver peça e se o tapete ou equipamento de destino não estiver alinhado ou preparado.
  - Ativação contínua do tapete exceder um período de tempo pré-definido sem passagem de peça.
- Tapete rotativo
  - Ativação do tapete para ambos os sentidos em simultâneo.
  - Ativação da rotação para ambos os sentidos em simultâneo.
  - Ativação do tapete para transferência de peça se o tapete tiver peça e se o tapete ou equipamento de destino não estiver alinhado ou preparado.
  - Ativação contínua do tapete exceder um período de tempo pré-definido sem passagem de peça.
  - Ativação contínua da rotação exceder um período de tempo pré-definido.
  - Ativação da rotação levar a exceder os sensores de posição de rotação.
  - Se com rotação ativa, ativação do tapete.
  - Se não alinhado, ativação do tapete.
  - Ambos os sensores de posição da rotação ativos (falha de *hardware*).
- Tapete deslizante
  - Ativação do tapete para ambos os sentidos em simultâneo.
  - Ativação do deslizamento para ambos os sentidos em simultâneo.
  - Ativação do tapete para transferência de peça se o tapete tiver peça e se o tapete ou equipamento de destino não estiver alinhado ou preparado. .
  - Ativação contínua do tapete exceder um período de tempo pré-definido sem passagem de peça.
  - Ativação contínua do deslizamento exceder um período de tempo pré-definido.
  - Ativação do deslizamento levar a exceder os sensores de posição de rotação.
  - Se com deslizamento ativo, ativação do tapete.
  - Se não alinhado, ativação do tapete.
  - Ambos os sensores de posição do deslizamento ativos.
- Máquinas de transformação
  - Ativação do movimento de um eixo para ambos os sentidos em simultâneo.

  - Ativação do movimento de um eixo levar a exceder os sensores de posição desse eixo.
  - Ativação contínua de um movimento exceder um período de tempo pré-definido.
  - Ativação da ferramenta se a mudar a ferramenta.
  - Ativação de ambos os sensores de posição de um eixo (falha de *hardware*).

- Armazém automático
  - No arranque, dois ou mais sensores do eixo X ou Z ativos (falha de *hardware*).
  - No arranque, todos os sensores do eixo Y com o mesmo valor (falha de *hardware*).
  - Ativação do movimento do eixo X ou Z levar a exceder os sensores de posição nos limites desse eixo.
  - Ativação do movimento de um eixo para ambos os sentidos em simultâneo.
  - Se o mecanismo de carga e descarga não estiver recolhido, movimentar eixo X.
  - Se não estiver alinhado com um local de armazenamento, alongar o mecanismo de carga e descarga na direção do armazém.
  - Se não estiver alinhado com os tapetes de carga e descarga do armazém, alongar o mecanismo de carga e descarga na direção dos tapetes.
  - Se alinhado com um local de armazenamento na sua posição baixa e com peça, alongar o mecanismo de carga e descarga.
  - Se alinhado com um local de armazenamento na sua posição alta e sem peça, alongar o mecanismo de carga e descarga.

- Sistema de montagem
  - Ativação do movimento de um eixo para ambos os sentidos em simultâneo.
  - Ativação do movimento de um eixo levar a exceder os sensores de posição nos limites desse eixo.
  - Dois ou mais sensores do eixo X ativos (falha de *hardware*).
  - Se não alinhado com os eixos X e Z, baixar.
  - Se o alinhamento dos eixos X e Z for sobre um tapete em movimento ou não alinhado, baixar.
  - Se a transportar peça, baixar num local ocupado por uma outra peça.
  - Se sem peça, fechar a garra.

- Pushers
  - Ativação do mecanismo para ambos os sentidos em simultâneo.
  - Ativação do movimento do mecanismo levar a exceder os sensores de posição.
  - Se a posição de descarga estiver cheia, ativação do mecanismo.
  - Se o tapete estiver em movimento, ativação do mecanismo.
  - Ativação contínua do mecanismo exceder um período de tempo pré-definido.

5.2.2.6 Implementação das proteções e identificação de proteções ativas

As proteções foram implementadas no *Codesys* como um *Program Organization Unit* (*POU*) do tipo *Function Block* (*FB*) para cada tipo de componente do sistema didático. Estes *FB* têm como finalidade ser instanciados dentro de outros *POU*, onde podem ser interligados e mapeados. Uma das principais vantagens deste tipo de implementação é que uma alteração no *FB* de um componente é replicada em todas as suas instâncias, tornando mais simples a correção de erros e alteração de funcionalidades.

Os *FB* de proteção tem como parâmetros de entrada valores do sistema didático e os valores para as saídas do sistema de controlo a serem avaliados. As saídas do *FB* são os atuadores do sistema didático e a informação das proteção ativas.

De seguida é apresentado parte do código desenvolvido para o *FB* de proteção do tapete linear:

```
(* Ativação do tapete para ambos os sentidos em simultâneo *)
IF (in_m_minus AND in_m_plus) THEN
lock_motor:=TRUE;
ELSE
lock_motor:=FALSE;
END_IF

(*Ativação do tapete para transferência de peça se com peça e o tapete ou
equipamento de destino não estiver alinhado ou preparado.*)
(* sentido menos*)
IF (in_m_minus AND in_a_m AND in_s_m) THEN
lock_a_m:=TRUE;
ELSE
lock_a_m:=FALSE;
END_IF
(* sentido mais
lock_a_p
*)
(* Ativação contínua do tapete sem passagem de peça exceder um período de tempo
pré-definido*)
timer_lock_time_motor
(IN:= ((in_m_minus OR in_m_plus) AND (NOT in_s_p)) ,
PT:= m_timeout, Q=> lock_time_motor, ET=> );

(* Ativação do motor*)
(*sentido menos*)
IF  (in_m_minus AND (NOT (stop OR lock_motor OR lock_time_motor OR lock_a_m))) THEN
out_m_minus:=TRUE;
ELSE
out_m_minus:=FALSE;
END_IF
(*sentido mais
out_m_plus
*)
```

De forma a implementar as proteções de todo o sistema didático foi criado um *POU* do tipo *FB* por cada célula (exemplo da célula de máquinas série na figura 5.3a), onde são instanciados os *FB* que representam os componentes da célula e feito o mapeamento das suas entradas e saídas.

Os *FB* de cada célula são instanciados num *POU* do tipo *Program* (figura 5.3b) a executar numa tarefa de alta prioridade com ciclo de 5 milissegundos. O tempo de ciclo escolhido visa garantir que entre os ciclos de comunicação com os módulos de entradas e saídas remotas, configurados para 10 milissegundos, é feita uma execução deste *POU*.

(a) Disposição dos *FB* dos componentes da célula máquinas série

(b) Disposição dos *FB* das células do sistema didático

Figura 5.3: Implementação dos *FB* de proteção

Os testes às proteções foram feitos a partir de painéis compostos por botões que atuavam diretamente as entradas dos *FB*, simulando o estado atual do sistema didático e dos comandos do sistema de controlo, e lâmpadas ligadas às saídas para validar o funcionamento, sendo os resultados os esperados.

5.2.2.7 Autenticação

De forma a limitar o acesso a algumas funcionalidades do *Interlock* foi implementada a autenticação de utilizador. A limitação pelo *Codesys* é feita pelas três opções de acesso de cada objeto de um painel: operável, apenas visível ou invisível, configuráveis para cada um dos quatro grupos de utilizador disponíveis, Nenhum, Operador, Serviço e Administrador.

Foram criados dois utilizadores, Demo a quem se deu acesso de Operador e Admin com acesso de Administrador, tendo sido dado os acessos por grupo:

- Nenhum - tem acesso à informação do estado do sistema didático, podendo ver os alertas atuais e o histórico mas não lhe é permitido fazer alterações ao funcionamento.
- Operador - tem a permissão adicional de poder alterar o modo de funcionamento do sistema didático de utilização remota para demonstração.
- Serviço - os mesmos acessos que Operador.
- Administrador - tem acesso a todas as funcionalidades disponíveis através dos painéis.

O controlo de acesso é feito por limitar as opções de acesso a um objeto que disponibiliza a ligação a um outro painel ou ativação de uma função que se deseja limitar.

5.2.2.8 Modos de funcionamento

O *Interlock* possui três modos de funcionamento, utilização remota, demonstração e administração.

O modo de funcionamento no arranque é o de utilização remota. Neste modo o sistema didático pode ser controlado remotamente por qualquer equipamento de controlo capaz de comunicar por *Modbus TCP/IP* e devidamente configurado.

O modo de demonstração desativa o controlo remoto passando a ser executado o programa de demonstração incorporado no *Interlock* . Durante o modo de demonstração as zonas de memória da utilização remota permanecem funcionais mas os seus valores são ignorados.

O modo de administração acede a painéis que permitem controlar diretamente os vários atuadores do sistema didático e ver o estado dos sensores, além de painéis que permitem alterar configurações e parâmetros do sistema didático.

5.2.2.9 Ligação ao semáforo do sistema didático

O semáforo instalado no sistema didático funciona a 24V DC, um valor de tensão que não normalmente possível de conectar diretamente a um *SBC*.

A base computacional anteriormente utilizada, a AP7000, estava equipada com uma placa de expansão com vários C.I. ULN2803, um *array* de oito transístores *Darlington*, o que lhe permitia fazer o controlo de dispositivos com diferentes tensões de funcionamento.

De forma a permitir à *Raspberry PI* controlar o semáforo e adicionar proteção elétrica às saídas foi implementado um optoacoplador seguido de um relé como representado na figura 5.4.

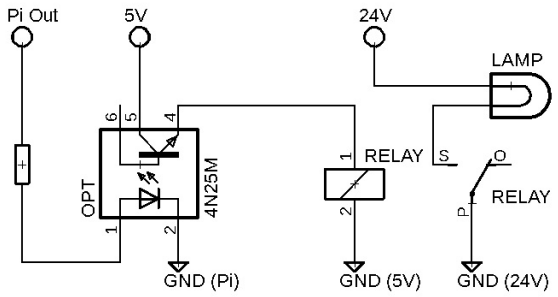


Figura 5.4: Esquemático do circuito adicionado ao GPIO da *Raspberry PI*

O funcionamento do circuito elétrico foi validado pela sua montagem numa *breadboard* e utilizando uma lâmpada de 12V como carga do relé, tendo sido verificado que a lâmpada acendia quando era ativa a saída da *Raspberry PI*. A versão final deste ponto da implementação engloba o desenho de uma placa de circuito impresso e a remoção das ligações elétricas do *Interlock* inicial, instalado no sistema didático até a validação final do projeto.

5.2.3 Resultados obtidos

As comunicações *Modbus TCP/IP* foram implementadas e testadas, tendo sido validado o seu funcionamento e mapeamento. A comunicação por porta série foi implementada e feitos testes ao seu funcionamento, tendo sido deixada a validação para a fase posterior de implementação dos escravos *Modbus RTU*.

O suporte para desenvolver e executar painéis foi garantido pelo *Codesys*, tendo sido testado e validado o seu funcionamento.

Após o levantamento e implementação das proteções não foram encontradas proteções em falta para implementar. As proteções implementadas foram testadas por painéis de teste, por onde as variáveis eram controladas, e pelo desenvolvimento de pequenos programas com erros propositados. O resultado dos testes foi o positivo, com o funcionamento desejado, prevenindo a ocorrência de situações que pudessem provocar danos.

A autenticação e os modos de funcionamento são geridos através do serviço de painéis, tendo sido validado o funcionamento correto dos acessos a funções selecionadas, como a alteração do modo de funcionamento apenas aos utilizadores especificados.

O circuito elétrico para controlo do semáforo foi validado e é funcional, faltando a implementação de forma a ser possível validar este ponto na totalidade.

5.3 Utilização de *Modbus RTU* entre Arduino e *Codesys*

5.3.1 Estudo de opções disponíveis

Para utilização do *Modbus RTU* entre periféricos que não o suportem e o *Codesys* é necessário a utilização de um conversor. Uma forma acessível de desenvolver um conversor, capaz de suportar vários protocolos e interfaces de comunicação de entrada como o I^2C, SPI ou UART, é recorrendo a um Arduino. De forma a integrar o *Modbus RTU* foram analisadas três opções de bibliotecas desenvolvidas para o Arduino para implementação do protocolo:

- Arduino-Modbus slave [73] - Suporte apenas para escravo *Modbus RTU*.
- modbus-arduino [2] - Suporte de *Modbus* mestre e escravo tanto por série (RS-232, RS-485) como *IP* via Ethernet. O suporte como mestre ainda em desenvolvimento.
- libmodbus [64] - Suporte para mestre e escravo *Modbus RTU*.

Os testes iniciais ao funcionamento das bibliotecas foram realizados com um Arduino Uno programado com os exemplos incluídos nas bibliotecas, conectado a um computador. Para verificação do funcionamento como escravo foi utilizada a ferramenta textitModbus Poll, um simulador de mestre *Modbus* já utilizado em testes anteriores ao escravo *Modbus TCP/IP*. Para verificação do funcionamento como mestre foram utilizadas as ferramentas *Modbus Slave* [66] e *Diagslave Modbus Slave Simulator* [63], simuladores de escravo *Modbus*.

Posteriormente foram feitos testes à adaptação do código das bibliotecas face às necessidades do projeto de forma a verificar a facilidade com que podem ser modificadas.

Os testes à biblioteca *Arduino-Modbus slave* mostraram que, apesar de ser a opção com a última atualização mais antiga, é capaz de funcionar corretamente como escravo e ser facilmente adaptável.

A biblioteca *modbus-arduino* foi apenas testada na vertente série em RS-232. O exemplo do modo escravo demonstrou funcionar corretamente mas a adaptação do código é bastante complexa face às restantes opções. O modo mestre não funcionou devidamente, não se tendo conseguido ler registos com a ferramenta *Modbus Slave* e com a ferramenta *Diagslave Modbus Slave Simulator* foi possível ler valores mas não se conseguiu fazer escritas.

A biblioteca *libmodbus* mostrou ser a ferramenta mais fácil de implementar e modificar, mas com limitações nas funções *Modbus* suportadas, apenas suportando registos. O modo escravo funcionou devidamente em placas Arduino Uno e Nano, contudo nas placas do braço robótico, derivadas do Arduino, a biblioteca mostrou não ser compatível. No modo mestre, de forma semelhante à biblioteca anterior, não se conseguiu obter os resultados esperados com ambas as ferramentas.

A partir destes resultados foi decidido apenas utilizar os Arduino como *Modbus RTU* escravo e escolhida a biblioteca *Arduino-Modbus slave* devido à sua compatibilidade com todas as diferentes placas utilizadas.

5.3.2 Implementação

Os parâmetros por omissão do *Codesys* para a comunicação *Modbus RTU* são de 10 milissegundos entre tramas e um *timeout* da resposta de 1 segundo. Visto não ser necessária uma frequência de atualização tão elevada, os parâmetros foram alterados para um intervalo de 300 milissegundos entre tramas, tendo sido mantido tempo para o *timeout* da resposta, de forma a reduzir a carga do processamento.

De forma a testar a comunicação do *Codesys* com escravos *Modbus RTU*, a partir da biblioteca *Arduino-Modbus slave* foi desenvolvido um programa que preenchia as várias posições de memória a serem acedidas por *Modbus RTU* a partir de um parâmetro definido no código a compilar, permitindo criar facilmente executáveis diferentes para distinguir os vários Arduino utilizados.

Os testes efetuados à comunicação com vários escravos *Modbus RTU* com o programa de teste demonstraram um funcionamento correto e sem falhas. A deteção do *timeout* da resposta foi verificada com dois testes, por *reset* ao Arduino e por fisicamente desconectar e conectar o escravo do *Raspberry PI*. O resultado dos testes foi semelhante, tendo sido observado que a comunicação entre o *softPLC* e o escravo falhava como esperado mas que não era reiniciada de forma automática apesar de configurada para tal (figura 5.5).

Figura 5.5: Configuração dos parâmetros *Modbus RTU*

Uma consulta mais pormenorizada no fórum do produto permitiu encontrar, além das configurações necessárias para ligação dos Arduino com a base computacional (5.2.2.4), uma forma de detetar falhas na comunicação nos dispositivos *Modbus RTU* e forçar o reinicio da porta de comunicação. Esta solução foi implementada através de um *POU* do tipo *Program* com um intervalo de execução de 1200 milissegundos onde é verificado o estado das portas série e em caso de falha é parada e reiniciada como demonstrado no extrato de código:

```
IF ((NOT(Modbus_Serial_Device_barcode.xAllSlavesOk)AND(cycle)))
THEN
Modbus_Serial_Device_barcode.xStop:=TRUE;
Modbus_Serial_Device_barcode.xResetComPort:=TRUE;
ELSE
Modbus_Serial_Device_barcode.xStop:=FALSE;
Modbus_Serial_Device_barcode.xResetComPort:=FALSE;
END_IF
```

A introdução deste *POU* permitiu passar os testes anteriores com sucesso mas sendo necessário alguns segundos para reiniciar a comunicação, entre um a três execuções do *POU*. As leituras feitas pelo escravo estão na sua memória e são atualizadas com o restauro da comunicação.

O teste ao número máximo de escravos *Modbus RTU* suportados foi feito com recurso a HUB USB alimentados para garantir que os requisitos elétricos não seriam limitados pela *Raspberry PI*. Os escravos, previamente configurados, foram fisicamente conectados ao HUB USB, tendo-se observado falhas na comunicação a partir de seis escravos conectados. A falha é semelhante a um *reset* ao escravo, aumentando a incidência das falhas com o aumento do número de escravos conectados. A mudança de HUB por outros modelos e a distribuição da carga por mais de um HUB não provocou alterações à ocorrência das falhas.

Esta falha nas comunicações não era prevista visto o limite de dispositivos USB da *Raspberry PI* segundo o encontrado no fórum de apoio do fabricante [20] é de 127 e no *Codesys* não é imposta nenhuma limitação ao adicionar portas série. Também foi identificada a incompatibilidade da *Raspberry PI* com alguns modelos de HUB USB, sendo detetada a conexão de um dispositivo USB mas sem o identificar.

Durante a implementação de outras componentes do projeto, o intervalo entre tramas e o *time-out* da resposta dos parâmetros da comunicação e o intervalo de execução do *POU* foram ajustados, estando estas alterações descritas nas respetivas secções (5.4 e 5.5).

5.3.3 Resultados obtidos

Foi possível validar o bom funcionamento entre múltiplos escravos *Modbus RTU* baseados em Arduino e o *softPLC*, assim como a capacidade de identificar falhas na comunicação e reiniciar as comunicações de forma autónoma após a correção da falha.

Um número alto de escravos mostrou criar falhas na comunicações e, apesar de o *softPLC* ser capaz de reiniciar as comunicações e o Arduino armazenar os valores dos registos, esta situação leva a atrasos significativos na comunicação e ao risco de alguns valores serem perdidos. De forma a evitar esta situação o número de escravos a utilizar deve ser limitado.

5.4 Expansão da identificação das peças por *RFID*

5.4.1 Estudo de opções disponíveis

Foram estudadas duas opções de implementação, uma para utilização das *tag RFID* atuais e outra para implementação de textittag referentes a outra norma.

A opção de manter as *tag RFID* existentes nas peças, que funcionam à frequência de 13,56MH respeitando a norma *ISO15693*, referida na secção 3.1.6, prevê adicionar leitores com suporte para a norma baseados no C.I. PN5180 [56] da NXP com protocolos de comunicação I^2C, SPI e UART para ligação a microcontroladores, com um custo aproximado de 10€ por leitor.

A outra opção estudada passa pela utilização de *tag RFID* baseadas na norma *ISO14443* [38, 39, 40, 41]. Os leitores encontrados para esta norma são baseados no C.I. RC522 [55], também da

NXP, e semelhantes aos leitores anteriores em aparência e protocolos de comunicação suportados, mas com um custo mais baixo, inferior a 2€.

O C.I. RC522 suporta a leitura e escrita de *tag*, tendo embutido *drivers* para ligação da antena conseguindo distâncias de operação até 50 milímetros, podendo ser utilizadas comunicações por I^2C, SPI e UART. As *tag* autocolantes para esta norma também têm um custo bastante reduzido, de aproximadamente 1€ por um conjunto de 10 *tags*.

Ambos os leitores cumprem os requisitos, pelo que a escolha foi a solução baseada na norma *ISO14443*, que apesar de necessitar adquirir novas *tag* para instalar nas peças é a solução mais económica.

5.4.2 Implementação

Os leitores adquiridos baseados no C.I. RC522 (figura 5.6), ao contrário do especificado, apenas disponibilizavam a comunicação por SPI. A alteração do protocolo nas placas é possível mas implica a remoção do C.I. para alterações nas pistas da placa, um processo delicado que, na ocorrência de um problema, pode levar à inutilização tanto do C.I. como da placa.

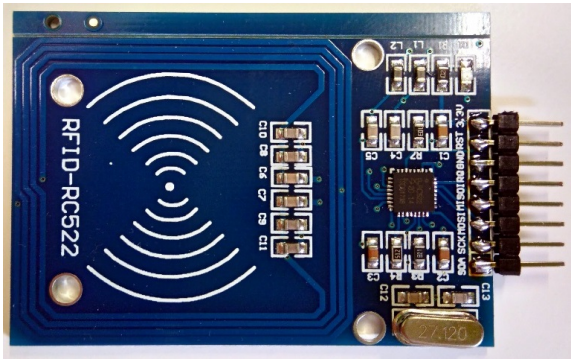


Figura 5.6: Placa do leitor baseado no C.I. RC522

Os testes iniciais foram realizados com recurso a uma biblioteca [49] desenvolvida para utilizar esta placa com um Arduino (figura 5.7) tendo sido validado o bom funcionamento do leitor e a sua capacidade de, colocado por baixo de um tapete, ler uma *tag* numa peça em movimento.

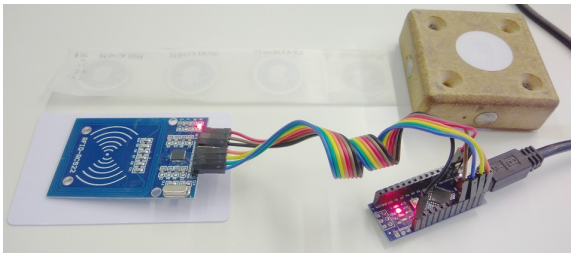


Figura 5.7: *Setup* de teste ao leitor *RFID*

A integração da biblioteca para funcionamento como escravo *Modbus RTU* foi realizada sem incompatibilidades, passando a ser a ser disponibilizados por *Modbus RTU* os dados obtidos pelos leitores.

De forma a ligar vários leitores a um único Arduino e dessa forma reduzir o número de microcontroladores necessários, foi utilizada uma outra biblioteca desenvolvida pelo professor Paulo Costa [7]. O limite de leitores suportados por esta biblioteca está relacionado com o número de pinos de saída disponíveis no microcontrolador, sendo necessárias duas saídas adicionais à comunicação por cada leitor.

No excerto de código abaixo é possível observar as ligações efetuadas entre quatro leitores e o Arduino. Entre os leitores o mesmo pino é ligado ao mesmo ponto do Arduino com a excepção dos pinos *MISO* e *NSS* em que é usado um pino do Arduino diferente para cada leitor.

```
/*****************************************************************
* RFID0  Arduino    RFID1  Arduino    RFID2  Arduino    RFID3  Arduino
* VCC    3.3V     = VCC    3.3V     = VCC    3.3V     = VCC    3.3V
* RST    2        = RST    2        = RST    2        = RST    2
* GND    GND      = GND    GND      = GND    GND      = GND    GND
* MISO   3        * MISO   7        * MISO   9        * MISO   11
* MOSI   4        = MOSI   4        = MOSI   4        = MOSI   4
* SCK    5        = SCK    5        = SCK    5        = SCK    5
* NSS    6        * NSS    8        * NSS    10       * NSS    12
* IRQ    NC       = IRQ    NC       = IRQ    NC       = IRQ    NC
*****************************************************************/
```

Os testes realizados a uma configuração de quatro leitores conectados a um Arduino Nano a funcionar como um módulo escravo *Modbus RTU* (figura 5.8) foram todos bem sucedidos.

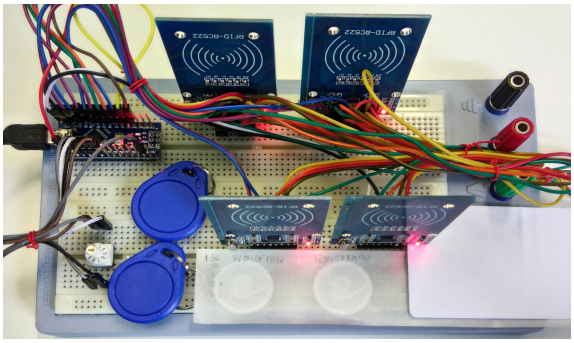


Figura 5.8: Teste ao funcionamento de quatro leitores *RFID* com *Modbus RTU*

Como o transporte das peças nos tapetes é lento, os parâmetros de comunicação foram novamente ajustados de forma a reduzir a carga no processamento. Os valores foram de 500 milissegundos de intervalo entre tramas e 1500 milissegundos para o *timeout* da resposta. O intervalo de execução do *POU* responsável por reiniciar as portas série em caso de falha também foi ajustado para 2000 milissegundos.

Testes posteriores à tolerância de mau alinhamento do leitor com a *tag* demonstram ser possível identificar as peças em movimento mesmo com uma diferença de 5 milímetros entre o centro do leitor e da peça.

5.4.3 Resultados obtidos

Os requisitos desta proposta foram cumpridos, tendo se obtido uma solução capaz de identificar peças em movimento nos tapetes, mesmo em situações não ideais.

A capacidade de ser possível ligar múltiplos leitores de *RFID* a um único Arduino garante uma redução do número de dispositivos a conectar ao *Raspberry PI*.

5.5 Modificação do sistema de identificação por código de barras

5.5.1 Estudo de opções disponíveis

Entre as opções de leitores de código de barras sem fios encontradas, as mais acessíveis funcionam como um um teclado USB. Existem também opções que permitem funcionar como uma porta série, algo que permitiria a troca direta com o leitor já existente, mas com um custo superior. Também é possível com *hardware* adicional fazer a conversão de teclado USB para porta série.

Manter o leitor implicaria a fixação do cabo ao longo da estrutura de suporte do sistema didático, e adquirir um suporte para o leitor capaz de recolher o cabo, de forma a que não houvesse o risco de se prender e danificar algum mecanismo. As opções de suportes e mecanismos de recolha de cabos encontradas tinham um custo próximo a um novo leitor sem fios.

Devido à alteração da arquitetura manter a interface do leitor existente não seria uma vantagem, pelo que sendo o custo de ambas as opções semelhante, foi optado fazer a mudança por um leitor sem fios. Um novo leitor não implica efetuar alterações na estrutura de suporte ao sistema didático, dando uma aparência mais moderna pela ausência de cablagem.

5.5.2 Implementação

Os testes para validação do novo leitor foram efetuados com o leitor conectado a um computador.

O alcance do sistema sem fios é satisfatório, permitindo fazer leituras em qualquer ponto do laboratório onde está situado o sistema didático. Também foi verificado que o estado da bateria do leitor afeta o alcance, reduzindo para 3 metros quando a bateria está próxima de necessitar de ser carregada.

Testes à leitura de vários códigos de caixas existentes no laboratório assim como códigos de teste impressos foram feitos com sucesso, sendo que o leitor apenas não foi capaz de ler códigos num ecrã, tanto de telemóvel como monitor de computador, mas que para a utilização desejada não é um requisito.

O *softPLC* não é capaz de obter diretamente os valores do leitor conectado à base computacional. A solução encontrada é semelhante à utilizada nos leitores *RFID*, recorrendo a um Arduino

Figura 5.9: Novo leitor de códigos de barras e conversor para *Modbus RTU*

a funcionar como um escravo *Modbus RTU*, com placa de expansão para suportar a ligação de um periférico USB [6, 50], convertendo os valores lidos para posições de memória *Modbus*. Esta solução é funcional e permite obter no *softPLC* os valores dos códigos lidos.

Ao repetir os testes efetuados para a validação do funcionamento *Modbus RTU* foi verificado que ao fazer *reset* ao Arduino a comunicação com o *softPLC* não era reposta. Foram repetidos os testes à comunicação com a ferramenta *Modbus Poll* que mostraram um bom funcionamento do conjunto na situação de *reset* do Arduino, sendo a comunicação resposta passado uns segundos. No caso de o Arduino ser desligado e conectado de novo não havia restauro de comunicação.

As diferenças na forma de comunicar encontradas entre o *Modbus Poll* e o *softPLC* eram o tempo de leitura entre tramas, 1 segundo ao invés de 500 milissegundos. A porta comunicação também não era reiniciada, sendo feitas múltiplas tentativas de comunicar com o escravo na situação de reset e perdendo o acesso à porta se o Arduino for desconectado.

As diferenças nos tempos levaram a se ajustar os parâmetros de comunicação para 1 segundo entre tramas, 2 segundos para o *timeout* da resposta e 2500 milissegundos para o intervalo de execução do *POU*. Os testes com estes parâmetros tiveram resultados positivos, sendo a comunicação reiniciada automaticamente.

5.5.3 Resultados obtidos

O leitor instalado permite identificar peças em toda a área do sistema didático, permitindo inclusive efetuar leituras alguns metros ao redor. Os resultados obtidos da solução implementada para conectar o leitor ao *softPLC*, após os ajustes dos parâmetros de comunicação *Modbus RTU*, foram igualmente satisfatórios, tendo os requisitos desta especificação sido cumpridos na totalidade

5.6 Painéis de informação e configuração do sistema didático

5.6.1 Estudo de opções disponíveis

O *Codesys* permite o desenvolvimento de painéis e o seu acesso remoto por página *WEB*, sem a necessidade de recorrer a outros recursos externos. A opção de uma consola industrial ou um sistema de *SCADA* foi descartada depois da escolha do *Codesys* como *softPLC* a utilizar.

5.6.2 Implementação

Os painéis podem ser gerados a partir de vários elementos simples como figuras geométricas, botões, interruptores, símbolos, lâmpadas e imagens, como elementos mais complexos como gestores de alarmes, calendário, caixas de mensagem e teclados virtuais. Também é possível aceder e controlar as várias variáveis internas do sistema, assim como criar e gerir listas de erros e alarmes.

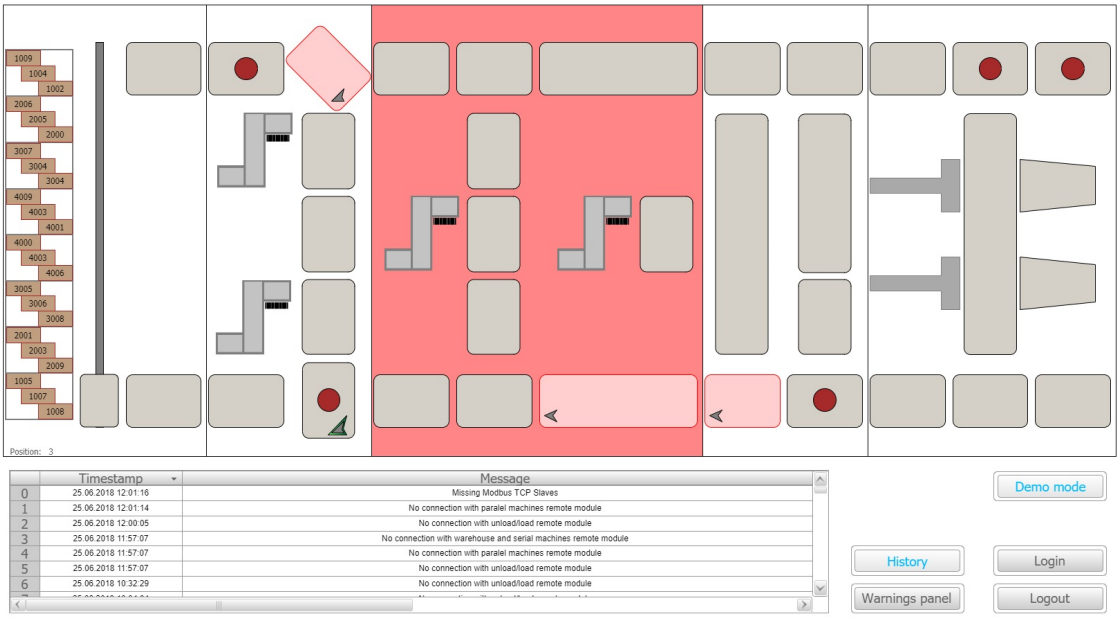


Figura 5.10: Painel principal

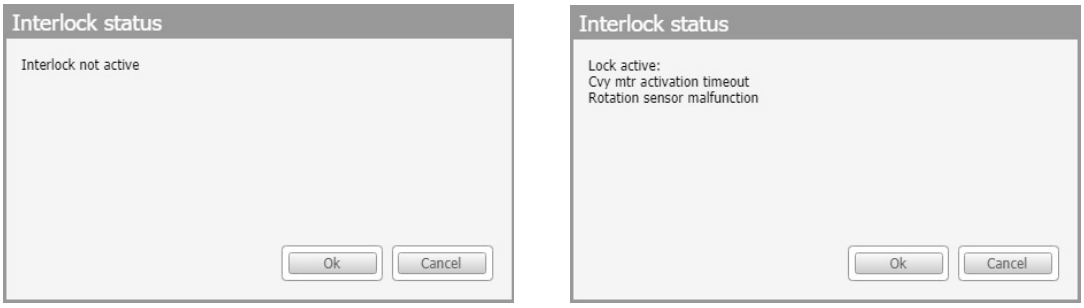
Na figura 5.10 está representado o painel inicial onde é apresentado o estado de todo o sistema didático. Uma célula em que não seja possível comunicar com o módulo de entradas e saídas remotas que a gere é colocada a vermelho (ver célula máquinas paralelas na 5.10), para informar da falha, sendo também adicionada uma entrada na tabela de alarmes.

O estado de funcionamento dos motores dos equipamentos é representado por setas cinzas, para indicar uma ordem do sistema de comando, e verdes para indicação que o comando está a ser processado (figura 5.11a). É possível observar que na figura 5.11b que apenas é visível a seta em cinza, indicando que há um comando para mover o motor mas que não foi atuado. Para indicação de uma proteção ativa o equipamento é colocado a vermelho.

Todos os equipamentos do painel inicial podem ser clicados, sendo apresentada uma caixa de texto a indicar a existência ou não de proteções ativas (figura 5.12).

(a) Funcionamento normal (b) Proteções ativas

Figura 5.11: Estado de funcionamento de um tapete rotativo



(a) Sem proteções ativas (b) Com proteções ativas

Figura 5.12: Caixas de mensagem com informação do estado das proteções de um tapete rotativo

No painel inicial é possível ver apenas 7 entradas em simultâneo da tabela de alarmes, sendo possível, através do botão *"Warnings panel"*, aceder a um outro painel que permite ver 40 entradas em simultâneo.

Foi também desenvolvido um painel de configuração para permitir a identificação das peças armazenadas (figura 5.14). Este painel é acessível clicando no armazém no painel inicial, disponível apenas a utilizadores autenticados.

Neste painel é possível carregar configurações de carregamento com valores pré-definidos de identificação, sendo o utilizador responsável por garantir que as peças existentes se encontram nos locais corretos, ou fazer o carregamento manual das identificações com recurso ao leitor de código de barras. O processo de carregamento manual consiste em ler o código da peça e depois no painel clicar na posição em que a peça se encontra, sendo-lhe atribuído o valor lido.

5.6.3 Resultados obtidos

Os painéis desenvolvidos no *Codesys* demonstraram serem capazes de cumprir todos os requisitos. O painel inicial permite ter conhecimento do estado de todo o sistema didático de forma direta, disponibilizando os detalhes das proteções ativas de um forma simples e intuitiva.

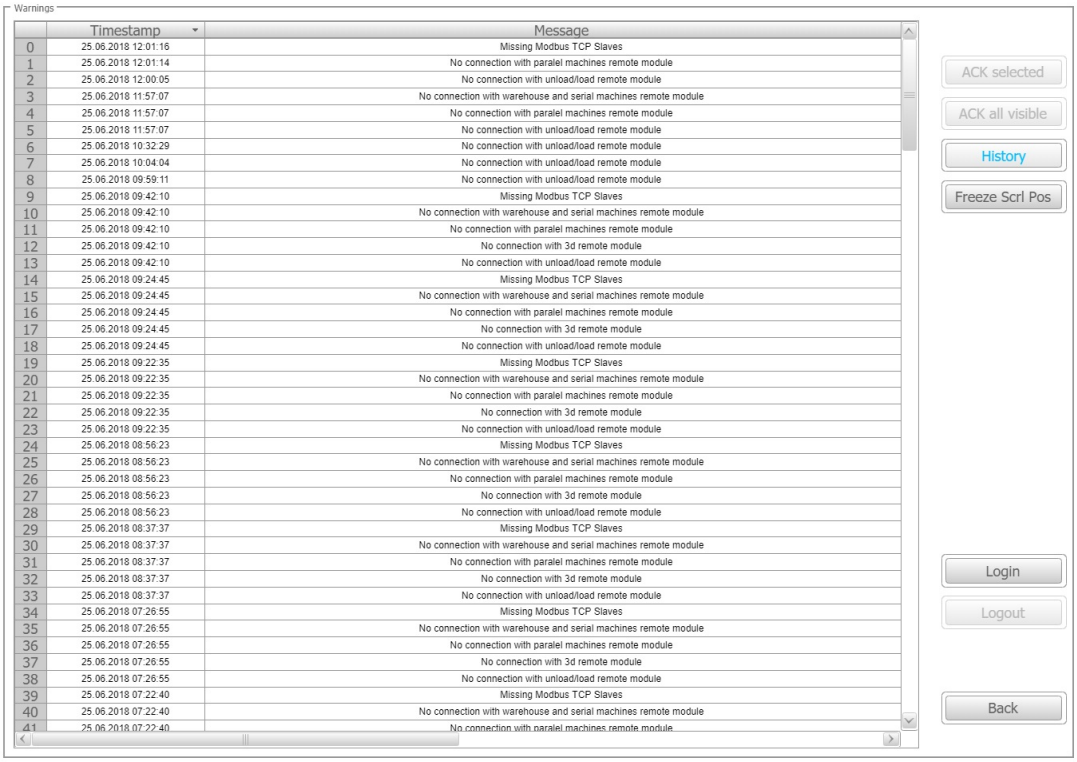


Figura 5.13: Painel de gestão da tabela de alarmes

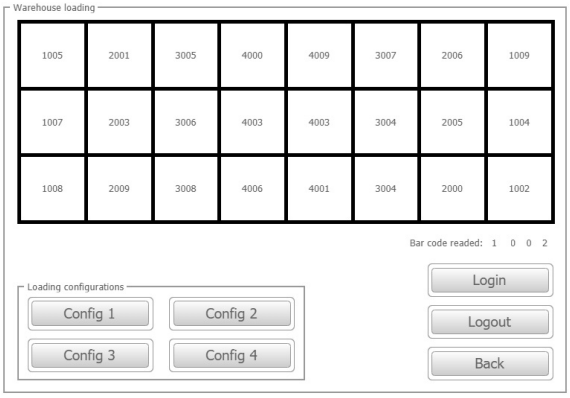


Figura 5.14: Painel de visualização e configuração da carga do armazém

5.7 Programa de demonstração inserido na base computacional

5.7.1 Estudo de opções disponíveis

A realização desta proposta implica que o novo programa de demonstração seja uma parte integrante do sistema didático. O *softPLC* e base computacional possuem recursos disponíveis suficientes para desenvolver o programa de demonstração, não havendo necessidade de utilizar um outro sistema de controlo.

Também é garantido pelo *softPLC* formas de programa de demonstração ser executado apenas por utilizadores autorizados, cumprindo todos os requisitos da proposta.

5.7.2 Implementação

O programa de demonstração utiliza todos os recursos das células do sistema didático, sendo desenvolvido de uma forma semelhante à implementação das proteções. Foi criado para cada componente do sistema didático um *POU* do tipo *FB*, a ser instanciado em *POU* do tipo *FB* que representam cada célula. Estes *POU* são depois instanciados no programa de demonstração, um *POU* do tipo *Program*.

Identificação e definição do trajeto das peças

As peças são retiradas do armazém para os tapetes superiores, sendo encaminhadas para uma das células seguintes, sendo depois encaminhadas para os tapetes inferiores para retornarem ao armazém (figura 5.15).

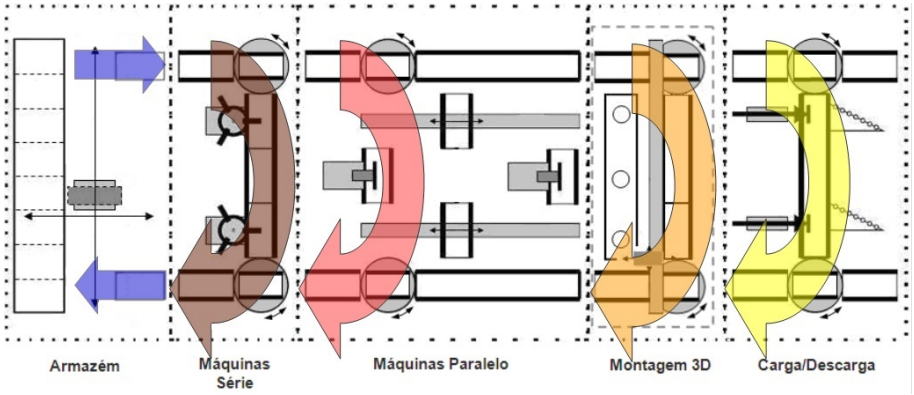


Figura 5.15: Trajeto das peças no programa de demonstração

O trajeto e transformação a realizar é definido pelo código de barras que identifica a peça. Os códigos das peças variam entre 1 e 4 para o primeiro dígito, utilizado para definir o destino e 0 a 9 para o último, utilizado para definir o processo a realizar.

As peças com identificação iniciada por 1 são encaminhadas para as máquinas série, por 2 para as máquinas paralelo, por 3 para a montagem e por 4 para a carga/descarga. Uma peça não identificada é encaminhada para primeira célula que passe seguindo o fluxo normal das peças. Uma peça com o valor para o processo de 0 é processada como não identificada em todas as células.

Na célula de máquinas série o valor da identificação para o processo é convertido na duração do tempo em que é feita a seleção e atuação da ferramenta.

Nas máquinas paralelo o valor para processo indica qual a máquina de destino e o tempo de atuação da ferramenta.

Na célula de montagem as peças são colocadas numa mesa de montagem disponível, ficando lá armazenadas por um tempo proporcional ao identificador de processo. As peças encaminhadas para a carga/descarga são enviadas pelos pushers para a primeira zona de descarga com capacidade, não havendo um processo diferenciado.

As peças que sejam colocadas no sistema sem terem sido previamente identificadas são reconhecidas como "0000", pelo que as peças devem ser identificadas antes de inicializar o programa de demonstração. A identificação é realizada através de um painel de configuração disponível no painel inicial, acessível apenas a utilizadores autenticados, já descrito na secção 5.6.

Lógica de funcionamento do transporte de peças

A lógica de funcionamento dos *FB* dos vários componentes para o transporte das peças (figura 5.16) é comum.

Quando um *FB* no estado **parado** recebe um pedido para receber muda para o estado **receção**. Caso seja colocada manualmente uma peça o estado passa para **ocupado**.

No estado **receção** é dada indicação que o pedido é aceite, ligando o motor e recebendo a informação da identificação da peça. Quando a peça termina de ter recebida é enviada a confirmação de peça recebida e o estado passa para **ocupado**.

No estado **ocupado** é feito um pedido ao *FB* seguinte para receber a peça, ficando à espera que seja aceite. Se aceite o estado passa para **envio**, se a peça for removida manualmente para o estado **parado**.

No estado **enviar** é ligado o motor e enviada a informação da identificação da peça até ser recebida a confirmação de recepção.

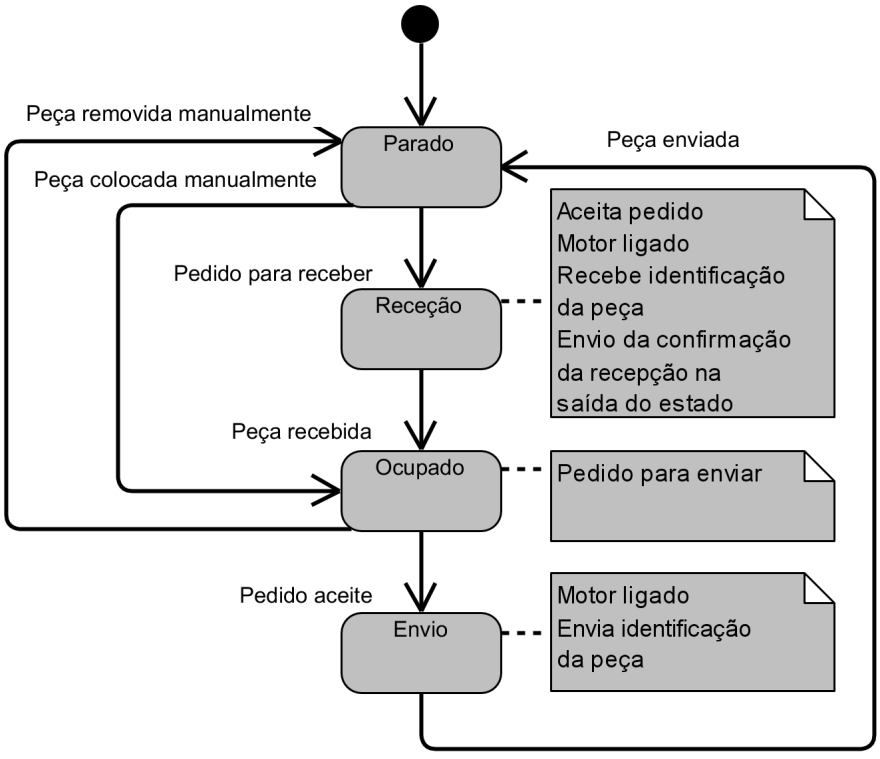


Figura 5.16: Diagrama da lógica de funcionamento do transporte de peças no programa de demonstração

Funções especificas dos componentes do sistema didático

As funções especificas de cada componentes são realizadas no seu *FB* no estado equivalente ao estado **ocupado** na lógica de funcionamento do transporte de peças, retomando depois as funções de transporte. As funções especificas de cada componente são:

- Tapete linear duplo - Ao receber uma peça transporta-a até ao sensor seguinte antes de fazer a chamada para o próximo *FB*. Quando utilizado num tapete linear simples o sensor de peça é ligado a ambas as entradas.

- Tapete rotativo - Ao receber uma peça verifica na sua identificação o destino. Se no destino, encaminha a peça para o interior da célula, caso contrário, encaminha para o próximo tapete. A recepção de peças pode ser feita por ambos os eixos, rodando para o eixo de onde veio o pedido.

- Tapete deslizante - Ao receber uma peça verifica na sua identificação a transformação e encaminha para a máquina indicada. Os pedidos de receção de peças podem partir de ambas as máquinas, alinhando o tapete com a máquina de onde veio o pedido.

- Máquinas de transformação - *FB* comum aos diferentes modelos de máquina que a partir do valor de transformação varia o tempo de atuação da ferramenta. A máquina é aproximada da peça, nos eixos disponíveis conforme o modelo, é atuada a ferramenta e depois retomada a posição mais afastada da peça.

- Máquina de montagem - Transporta as peças de um tapete da célula para uma mesa de montagem disponível para serem processadas. Uma peça depois de processada informa a máquina de montagem que a retira da mesa novamente para o tapete para ser transportada para fora da célula.

- Pushers - Se um espaço de descarga estiver disponível, a peça é encaminhada para o pusher correspondente e empurrada para o espaço de descarga. Se ambos estiverem ocupados, a peça fica em frente do primeiro pusher até um local de descarga estar disponível.

- Armazém - Se o tapete para retirar peças estiver livre, é retirada a peça armazenada no local de armazenamento ocupado com identificação mais alta. Se o tapete de retirar estiver ocupado ou o armazém sem peças, uma peça que esteja no tapete para depositar peças é colocada no armazém no local de armazenamento disponível com identificação mais baixa.

Implementação dos *Function Blocks*

Na figura 5.17 é possível observar o *Sequential Function Chart* (*SFC*) do *FB* que implementa a lógica de funcionamento do um tapete linear e as semelhanças com o diagrama da 5.16.

Para cada célula foram interligados os *FB* correspondentes aos seus componentes, estando na figura 5.18 representada a composição do *FB* de controlo da célula máquinas série.

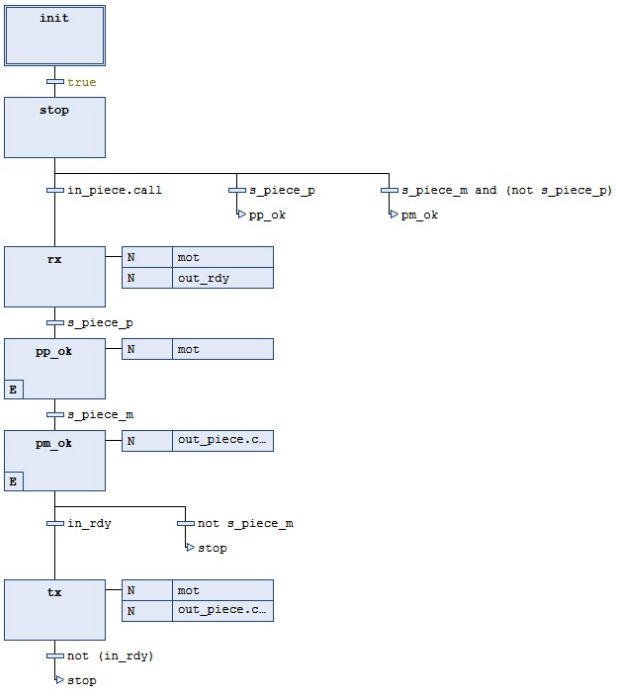


Figura 5.17: *Function Block* de controlo de um tapete linear

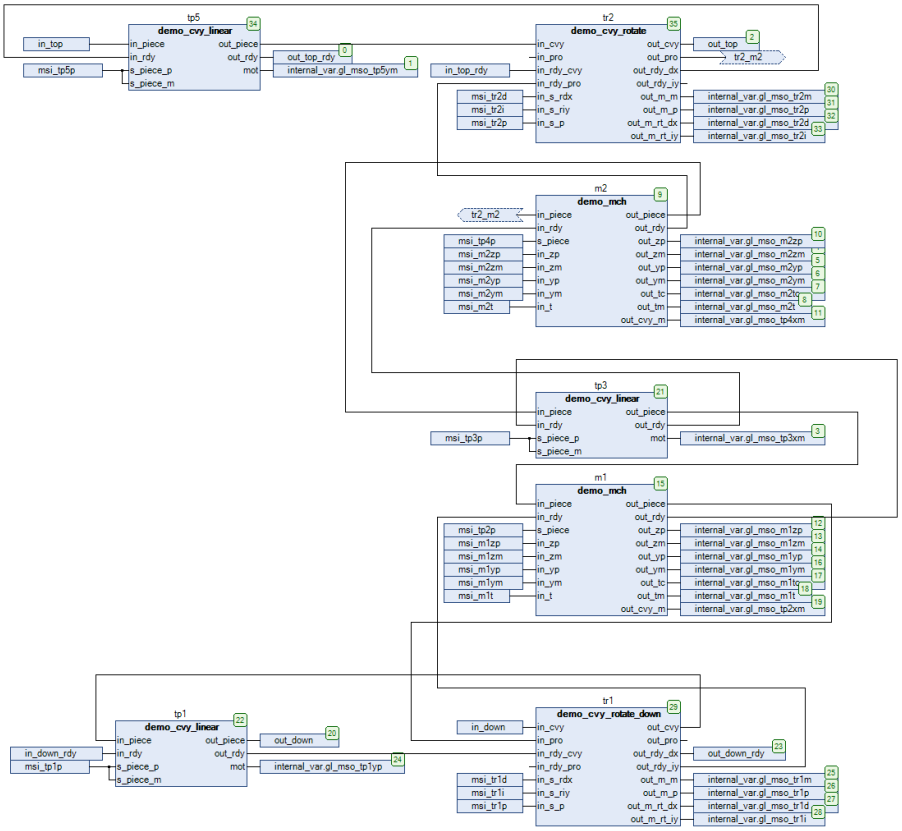


Figura 5.18: *Function Block* de controlo da célula máquinas série

5.7.3 Resultados obtidos

O programa de demonstração cumpre todos os requisitos, sendo executado diretamente no *Codesys* e por utilizadores autorizados. Os *FB* desenvolvidos para o programa de demonstração foram pensados a prever a retirada de peças durante os processos, não provocando situações de bloqueios na execução do programa.

5.8 Correção da limitação mecânica da célula de montagem

5.8.1 Estudo de opções disponíveis

A existência de uma limitação mecânica no eixo *Z* do sistema de transporte desta célula leva a que o movimento de elevar peças seja feito com dificuldade, levando ao longo do tempo a que o motor responsável por este movimento avarie.

Das opções referidas na secção 4.3.1 foi decidido implementar a redução de atritos e a substituição do motor.

A procura de pontos de atrito anómalos, verificando todos os pontos de contacto do sistema de transporte ao ser movimentado o eixo *Z* permitiu identificar pontos de atrito elevados no *encoder* responsável por ler o posição deste eixo.

A substituição do motor implica que para garantir a compatibilidade da montagem é desejada a utilização de material do fabricante. O motor responsável pelo movimento do eixo é da *Fischertechnik* com de 2,4W de potência, com o *Part number* FI-75245 (figura 5.19a), tendo sido encontrado no catalogo um motor de 7,2W com o *Part number* FI-37488 (figura 5.19b).

(a) Motor original

(b) Motor de substituição

Figura 5.19: Motores *Fischertechnik*

As opções de instalar um contra-peso de forma a ajudar o movimento ascendente e redução de peso da parte do sistema de transporte foram descartadas. Ambas as opções seriam de difícil implementação e teriam pontos negativos como o peso extra do contra-peso prejudicar o movimento dos restantes eixos ou a redução do peso que iria reduzir a integridade física do mecanismo.

5.8.2 Implementação

O volume do novo motor implicou a desmontagem parcial do módulo de transporte para alterações na estrutura de forma a permitir a sua instalação.

Após a instalação o motor foi testado, alimentando diretamente os terminais do motor, tendo se validado a capacidade para elevar a garra. A instalação do motor terminou com a montagem do módulo de transporte de forma a manter o seu aspeto inicial e refazer o teste à capacidade do motor, comandado pelos módulos de entradas e saídas remotas.

O atrito excessivo no *encoder* tinha origem no suporte do *encoder* que permite a sua fixação nas peças da *Fischertechnik*. O problema foi corrigido pelo aumento do diâmetro do suporte na zona do veio. Depois de retificado o diâmetro foi possível verificar que o movimento do *encoder* passou a ser livre como esperado.

5.8.3 Resultados obtidos

Após a substituição do motor e remoção do atrito no *encoder* o movimento passou a ser feito sem falhas, sendo capaz de elevar três peças num tempo próximo ao de efetuar o movimento de elevação sem carga, demonstrando que o movimento é feito sem esforço.

5.9 Máquinas de simulação de processos para a célula de montagem

5.9.1 Estudo de opções disponíveis

As máquinas existentes simulam um processo de transformação recorrendo a um motor para girar uma roda dentada. As dimensões deste mecanismo não são compatíveis com os locais onde se deseja instalar as novas máquinas.

A opção idealizada é a simulação de um processo físico, como um forno ou exposição a luz, através de *Light-emitting diode* (*LED*) *red, green, and blue color model* (*RGB*) colocados numa placa de pequenas dimensões, que pode ser facilmente instalada nos locais desejados.

O controlo de luminosidade de um *LED* é feito com recurso a *Pulse Width Modulation* (*PWM*), sendo no caso de um *LED RGB* necessário um *PWM* por cor. O controlo por um microcontrolador como o ATMEGA328P existente no Arduino Uno ou uma base computacional como a *Raspberry PI* é possível mas limitado, visto não possuírem muitos pinos com a suporte para *PWM*.

A forma encontrada de contornar essa limitação foi pela utilização de uma placa de expansão, baseada no componente PCA9685 da NXP [36, 54], que permite controlar 16 sinais de *PWM*. O comandado é realizado por I^2C, protocolo de comunicação suportado tanto por Arduino como a *Raspberry PI*.

Numa fase posterior do projeto também foi verificada a compatibilidade da placa de expansão com o *softPLC* escolhido, podendo ser controlado diretamente por ele.

5.9.2 Implementação

Devido a problemas logísticos não foi possível obter a placa de expansão a tempo de ser implementada no projeto. Para validação do conceito foi desenvolvida uma versão da máquina, baseada em Arduino Nano, a funcionar como um escravo *Modbus RTU*, com os *LED* controlados diretamente por saídas do Arduino. Visto existirem seis pinos com suporte para *PWM* num Arduino Nano, apenas é possível controlar 2 *LED RGB* por escravo, muito abaixo dos 5 suportados com recurso à placa de expansão.

5.9.3 Resultados obtidos

Os resultados obtidos nesta parte da implementação apesar de permitirem validar o funcionamento e controlo das máquinas a partir do *softPLC*, a impossibilidade de proceder à implementação com a placa de expansão limitou os resultados esperados, não tendo sido conseguido criar o protótipo desejado.

5.10 Informação do estado de ocupação do Armazém

5.10.1 Estudo de opções disponíveis

O estudo das opções desta funcionalidade é dividido em duas secções, a escolha do sensor utilizado para detetar as peças e a forma como a informação do sensor é transmitida ao *Interlock*.

5.10.1.1 Sensor

Para deteção das peças foram estudados três tipos de solução: baseado em visão, em sensores indutivos e em sensores ópticos.

O sensor baseado em visão consiste na analise de imagens de forma a detetar a existência de peças no armazém. A aquisição das imagens é usualmente feita por uma câmara de vídeo ou periférico equivalente. A *Raspberry PI* suporta sistemas de visão, tanto do fabricante (figura 5.20) como *webcam* ligadas por USB [18, 21], e a execução do *OpenCV* [68], uma biblioteca *open source* multi-plantaforma para *computer vision* e *machine learning*, permitindo definir e encontrar padrões.

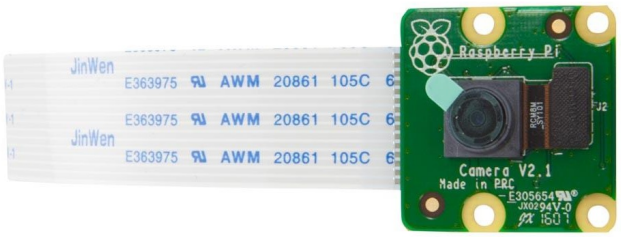


Figura 5.20: Câmara desenvolvida para a *Raspberry PI*

Esta opção tem a vantagem de permitir com um único sensor, montado no exterior do armazém e não influenciando o seu funcionamento, obter a informação da ocupação. As desvantagens são os requisitos computacionais, que limitariam os recursos disponíveis na *Raspberry PI*, e a sensibilidade à luminosidade e cor, bastando pequenas alterações na luz ambiente ou na posição do sistema didático no laboratório para o sensor deixar de funcionar devidamente e ser necessária uma nova calibração.

Os sensores indutivos detetam objetos metálicos através de um campo magnético, não detetando outro tipo de objetos. Este é o tipo de sensor mais utilizado no sistema didático para detetar as peças em movimento nos tapetes. O modelo existente tem uma distância de deteção de objetos de aço de 4 milímetros. O funcionamento deste sensor é muito satisfatório mas devido às suas dimensões não é possível o instalar num local onde esteja alinhado com os pontos metálicos das peças. As alternativas encontradas com dimensões compatíveis não possuíam uma distância de deteção suficiente para deteção as peças.

Os sensores ópticos são baseados num conjunto de emissor e receptor do mesmo tipo de luz, havendo três tipos principais de sensores:

- Barreira - O emissor e recetor estão separados, sendo colocados frente-a-frente e detetando objetos que interrompam o feixe. Tem a vantagem de detetar todos os objetos cuja composição bloqueie a passagem da luz vinda do emissor mas a ligação elétrica mais complexa pelo facto de o emissor e recetor necessitarem de ser alimentados.
- Reflexão - O emissor e recetor estão juntos, simplificando a ligação elétrica, sendo necessário um refletor em frente do sensor para devolver a luz, sendo o restante funcionamento semelhante ao de barreira.
- Difusão - O emissor e recetor estão juntos, sendo a reflexão feita no objeto a detetar. A vantagem deste tipo é de apenas ser necessário o sensor, contudo as distâncias de deteção são mais curtas que os outros tipos e a qualidade da deteção está ligada à cor e material do objeto a detetar.

(a) Barreira (b) Reflexão (c) Difusão

Figura 5.21: Sensores ópticos

As soluções encontradas de sensores ópticos com eletrónica para alimentação e saída inclusa não tinham dimensões compatíveis e/ou eram dispendiosas, tendo sido procuradas soluções baseadas unicamente em *LED* emissor e fototransístor.

Testes feitos a uma solução do tipo difusão com o sensor APDS-9104 [1], baseado num *LED* emissor e fototransístor montados num suporte não foram satisfatórios. O sensor era capaz de detetar uma etiqueta branca num alcance de 1 a 7 milímetros com intensidade suficiente para atuar a entrada, contudo não foi possível detetar uma peça do sistema didático.

A *datasheet* do fototransístor indica que uma passagem de corrente de coletor superior a 80% entre distância de deteção de 2 a 6 milímetros mas sem indicar o tipo de superfície. Pela visualização de *datasheet* de sensores semelhantes é possível concluir que os valores são indicados para uma superfície ideal branca, sendo que a corrente da saída desce com cores mais escuras e superfícies pouco refletoras.

Testes a uma solução com o emissor TSAL4400 [72] e recetor OP505 [70] numa configuração do tipo barreira mostrou ser funcional, tendo se conseguido uma alcance de deteção superior a 8 centímetros. O custo de cada conjunto de emissor e receptor é inferior a 1€, sendo sido a opção de sensor escolhida para implementação.

5.10.1.2 Transmissão de informação ao *Interlock*

O *Interlock* obtém as informações do estado dos sensores do sistema didático pelos módulos de entradas e saídas remotas, sendo utilizadas as entradas físicas da base computacional para a informação vinda do controlador de emergência. Nem os módulos de entradas e saídas remotas nem a base computacional têm entradas disponíveis suficientes para o número de sensores a adicionar, sendo necessária uma alternativa.

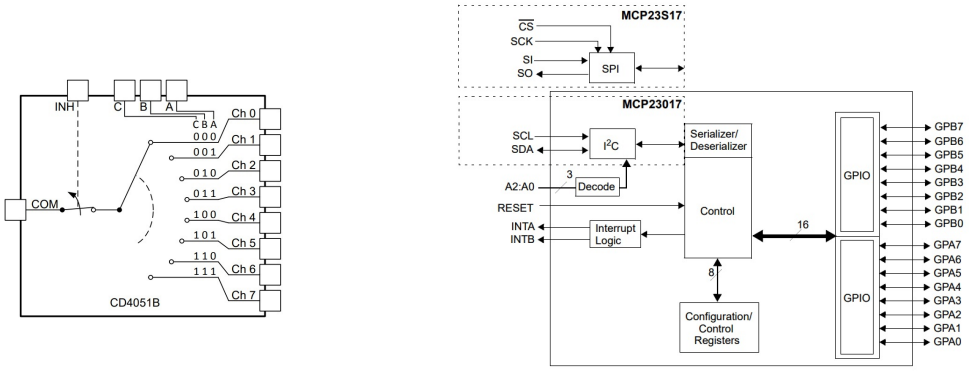
Os módulos de entradas e saídas remotas são expansíveis, sendo possível adquirir cartas de entrada adicionais mas o seu custo é significativo, aproximadamente 250€ para um módulo de 16 entradas, não sendo uma opção devido ao custo elevado.

A expansão das entradas da *Raspberry PI* é possível através da utilização de multiplexadores e *I/O Expander*. Um multiplexador consiste num C.I. com um funcionamento semelhante a um seletor, ligando a entrada escolhida à sua saída. O *I/O Expander* consiste num C.I., comandado por um protocolo de comunicação como SPI ou I^2C, com um funcionamento semelhante aos pinos de entrada e saída de um microcontrolador. Podem ser configurados como saídas ou entradas, com ou sem resistência de *pull-up*.

A opção baseada em multiplexador estudada foi o C.I. CD4051 da *Texas Instruments* [69] (figura 5.22a). Este multiplexador tem um custo reduzido, inferior a 0,5€. Permite fazer a comutação de oito entradas, necessitando de três saídas do microcontrolador para selecção do canal, podendo as mesmas três saídas utilizadas para controlar mais do que um multiplexador.

Como opção de *I/O Expander* foram tidos os MCP23S17 e MCP23017 da *Microchip* [52] (figura 5.22b). Estes *I/O Expander* apenas variam no protocolo de controlo, por SPI e I^2C respetivamente, permitindo configurar individualmente 16 pinos de entrada e saída com um custo inferior a 1,2€.

O custo por entrada das opções são próximos, assim como a quantidade de ligações eletricas entre o sistema de controlo e a opção para expansão, tendo sido escolhido as opções de *I/O Expander* porserem suportadas pelo *Codesys*.



(a) Multiplexer CD4051 (b) *I/O Expander* MCP23017/MCP23S17

Figura 5.22: Diagrama de funcionamento das opções de expansão

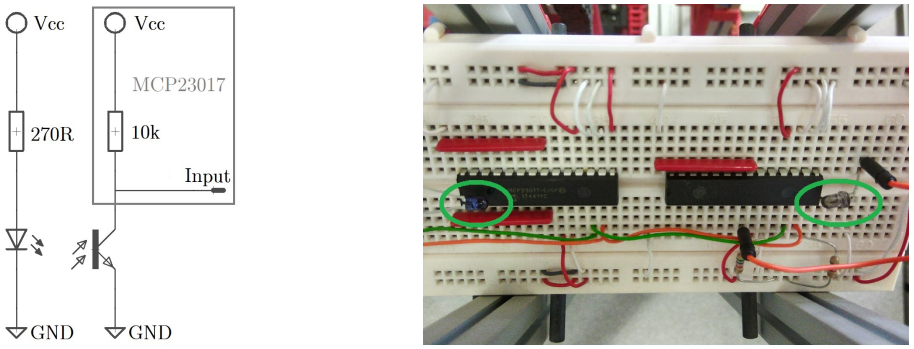
5.10.2 Implementação

Para permitir a implementação dos *I/O Expander* foi necessária a ativação dos protocolos na base computacional, desativados por omissão, executando a configuração do sistema com o comando *"raspi-config"*, e instalação das bibliotecas dos componentes no *Codesys*. Testes feitos a partir do *Codesys* para leitura de entradas e atuação de saídas em ambos os modelos de *I/O Expander* não tiveram sucesso apesar de no *Codesys* ser indicado a comunicação do protocolo como funcional.

Para verificar a possível existência de uma avaria nos *I/O Expander* foram realizados os mesmos testes, sendo o controlo realizado com um Arduino, tendo se confirmado que os C.I. estavam funcionais, tendo sido possível a configuração dos pinos como entradas com e sem resistência de *pull-up* e como saídas.

Para validação do funcionamento com o *I/O Expander* MCP23017 foi ligando o fototransístor diretamente a um pino de entrada do MCP23017, utilizando a resistência interna de *pull up* de 10kΩ para alimentar o coletor do fototransístor (figura 5.23a).

Testes realizados à solução integrada num local de armazenamento (figura 5.23b) validaram que a distância de deteção cumpria os requisitos, sendo validada a solução para implementação.



(a) Esquema eletrico do sensor (b) Implementação com o *I/O Expander*

Figura 5.23: Montagem de teste para validação do sensor ótico com *I/O Expander*

5.10.3 Resultados obtidos

Os resultados da utilização do sensor optico de barreira composto nos componentes TSAL4400 e OP505 com o *I/O Expander* MCP23017 são satisfatórios. A solução tem um custo bastante reduzido, ficando os 24 sensores e dois *I/O Expander* abaixo de 25€.

O protótipo desenvolvido necessita de um Arduino a funcionar como escravo *Modbus RTU* para fazer o interface entre o *Interlock* e os *I/O Expander*, contudo há a possibilidade de no *Codesys*, caso não seja possível descobrir a falha nos testes realizados, desenvolver uma biblioteca de raiz para permitir a ligação direta com o MCP23017 ou MCP23S17.

5.11 Desenvolvimento de célula baseada num braço robótico

5.11.1 Estudo de opções disponíveis

Numa primeira fase foram avaliados braços que pudessem cumprir os requisitos, tendo sido excluídos modelos cujo controlo não fosse elétrico e o raio de ação suficiente. A partir dessa pesquisa foram selecionados alguns braços para análise mais detalhada da capacidade em cumprir os restantes requisitos. Os braços selecionados foram o Owi 535 [8, 33], Lynxmotion AL5D ,em diferentes opções [34, 61], e o *uStepper Robot Arm* [45, 47], que podem ser visualizados na figura 5.24. Na tabela 5.3 é possível observar características destes braços.

(a) Owi 535 (b) Lynxmotion AL5D (c) uStepper Robot Arm

Figura 5.24: Braços robóticos analisados

O Owi535 não possui informação da abertura do *gripper*, não sendo por isso possível validar este campo. O facto de utilizar um comando de motor DC direto também invalida a sua utilização pois este tipo de controlo não garante precisão de posição.

O Lynxmotion AL5D, na sua vertente com motores e eletrónica, e o *uStepper Robot Arm* podem ser utilizados com a alteração do *gripper* para um de maior abertura.

O Lynxmotion AL5D, graças ao interface da placa de controlo, é facilmente controlável através de simples comandos por RS232, contudo este controlo é feito em malha aberta e sujeito a eventuais erros.

O *uStepper Robot Arm* usa um controlo em malha fechada de motores de passo que apesar de não permitir um controlo tão simples como o AL5D possuem uma maior precisão pelo que se torna uma melhor escolha.

Tabela 5.3: Características dos braços robóticos

		Owi535	AL5D	AL5D PLTW	uStepper
Raio de alcance (cm)		32	26	26	42
Graus de liberdade		4	4	5	4
Gripper	Abertura (cm)	Sem info	3.2	3.2	4
	Alteravel	Não	Sim	Sim	Sim
Interface de comando	Local	Sim	-	-	-
	RS232 TTL	-	Sim	Sim	
	USB	Opção	Sim	Sim	Sim
	Bluetooth	-	Opção	Opção	Sim
Comando motor		Direto	Sem motores	Servo	Passo
Fonte de alimentação		4x Pilhas D	6V, 3A	6V, 3A	19V, 4.7A
Preço (sem IVA, em €)		55 (+45 módulo USB)	170 (+ 120 motores +35 eletrónica + 45 pulso)	370 (com placa de comando e software)	370

Neste contexto a opção escolhida foi o *uStepper Robot Arm* por ser a solução que cujos motores e respetivo controlo permitem uma maior precisão, aliado ao facto de ser disponibilizado para edição e impressão 3D os componentes da estrutura do braço, o que permite criar peças para reparação e alteração de características como a abertura do *gripper*.

5.11.2 Validações iniciais e planeamento da implementação

5.11.2.1 Montagem e validação mecânica do braço robótico

O *uStepper Robot Arm* é adquirido em peças, pelo que o primeiro passo da implementação passou pela montagem e ajustes de afinação, segundo as instruções [46], do braço robótico.

O braço é composto por três eixos de movimento, um para rodar sobre a sua base e dois para variar a altura e alcance do *gripper*. Cada eixo é atuado por um motor de passo. A estrutura do braço garante que o o *gripper* é mantido na horizontal, independentemente da sua posição, e é atuado por um servomotor.

Para verificação do funcionamento da mecânica do braço foram feitos de forma manual testes a todos os movimentos possíveis, tendo sido encontrados pontos de atrito em algumas engrenagens e rolamentos devido a mau alinhamento na montagem. Após correção do alinhamento foram refeitos os testes tendo sido validada a mecânica do braço. Na figura 5.25 estão representados os eixos de movimentos do braço.

5.11.2.2 Validação do *hardware*, biblioteca e controlo

O controlo do braço é realizado por placas instaladas na base dos motores de passo (figura 5.26). Estas placas são baseadas num microcontrolador compatível com Arduino, tendo como periféricos uma *drive* para controlo do motor e um *encoder* para deteção da posição do eixo.

A validação do *hardware* e controlo foi realizada recorrendo à biblioteca do fabricante [48] e ao programa de demonstração fornecido. A arquitetura do braço definida pelo fabricante é de um

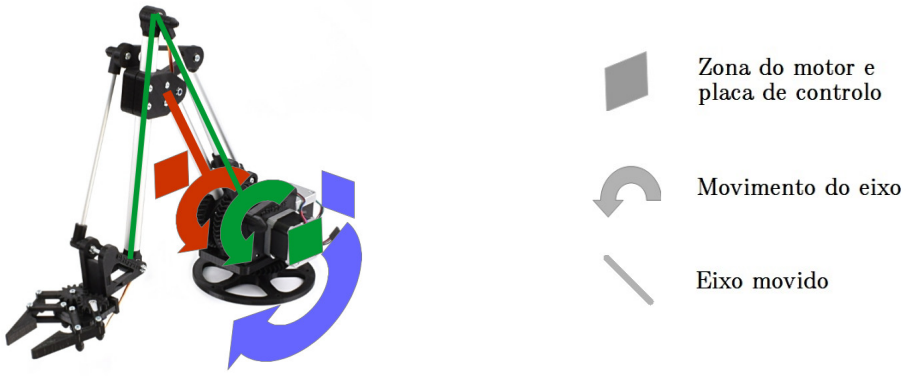


Figura 5.25: Eixos de movimento do *uStepper Robot Arm*

mestre e dois escravos, sendo necessário interligar as placas de controlo do braço (figura 5.27) de forma a permitir a comunicação entre as placas.

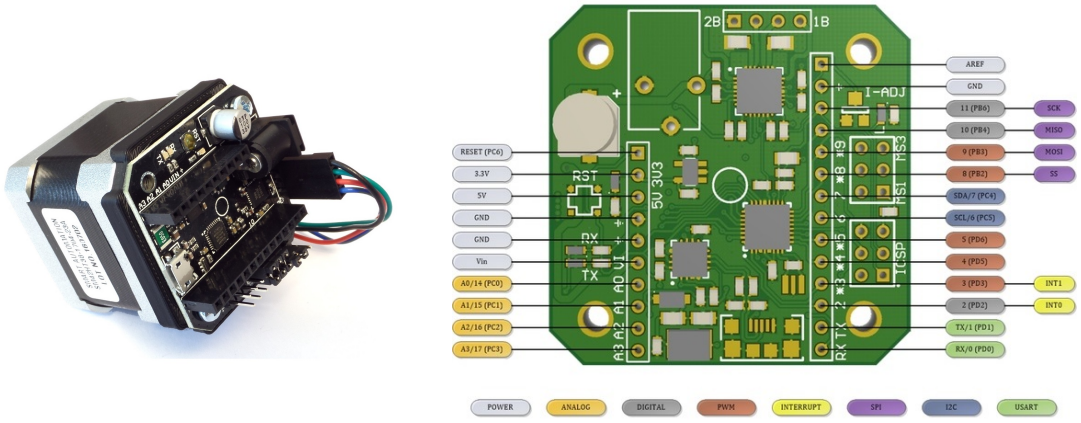


Figura 5.26: Placa de controlo *uStepper*

O programa de demonstração tem como funcionamento gravar movimentos feitos ao braço de forma manual e depois os reproduzir autonomamente. Testes recorrendo a este programa permitiram verificar a capacidade do braço para transportar objetos com peso semelhante às peças do sistema didático. Também foi verificado que quando eram feitas gravações de movimentos longos, próximos de uma rotação do motor, a trajetória reproduzida não era a esperada. Após depuração do código não foi possível identificar o *bug*, podendo ser originado por erros da biblioteca ou da falta de implementação do *feedback* sobre o movimento real do motor face ao comandado.

A análise à biblioteca e às funções disponíveis foi realizado pelo desenvolvidos vários programas para testar as suas funcionalidades, tendo sido encontrados *bugs* em algumas funções que utilizam dados do *encoder*, necessárias para o controlo direto em malha fechada.

A alternativa encontrada passa por utilizar as funções de controlo do motor em malha aberta, que não recorrem ao *encoder*, lendo o valor do *encoder* antes e depois do movimento para verificação do movimento realizado. Testes mostraram numa situação de movimento sem falhas (sem deslizamentos mecânicos) um erro entre o ângulo movido e medido de 2%.

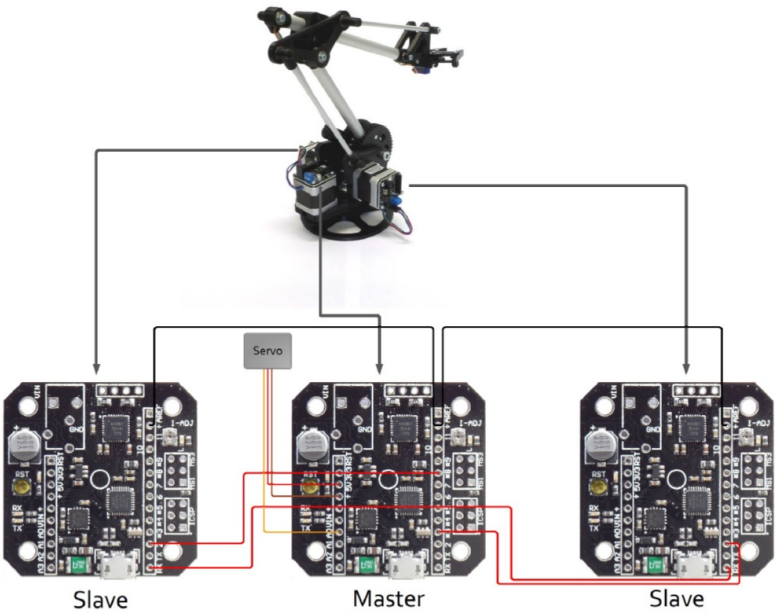


Figura 5.27: Ligações entre placas do *uStepper Robot Arm* para o programa de demonstração

5.11.2.3 Planeamento da arquitetura a implementar

Depois de feitos os testes à mecânica, *hardware*, biblioteca e controlo do braço foi projetada a arquitetura a implementar no mestre e nos escravos. Devido aos *bugs* encontrados nos testes das funções da biblioteca, foi decidido nesta fase fazer um controlo em malha aberta, sendo as ordem de movimento dos motores dadas no ângulo a mover.

A arquitetura a implementar no mestre é representada na figura 5.28) e composta pelos seguintes blocos:

- **Proteção** deteta se o eixo a ser comandado atingiu os limites físicos definidos, parando o motor sem travão (com movimento livre). O estado de intervenção deste bloco é indicado no estado do sistema.
- **Arranque** verifica se o funcionamento do *hardware* é o esperado. O resultado da verificação é indicado no estado do sistema e se com sucesso é ativo o bloco de **gestão** e **garra**, caso contrário o braço é bloqueado até ser feito um *reset*.
- **Comunicação com exterior** é responsável pela receção das instruções de comando vindas do exterior e fornecer informação do estado do braço. O bloco **comunicação com escravos** tem como função o envio de comandos para os escravos e receber a informação do seu estado.
- **Garra** é responsável pelo *gripper* do braço garantindo as condições de funcionamento do servomotor que o controla.
- **Controlador motor** é responsável por fazer o interface entre o controlo e a potência e obter as informações de estado relacionadas com o motor.

- **Comunicação com escravos** envia os comandos vindos da **gestão** e fornece informações do estado dos escravos.

- **Gestão** Verifica se as instruções recebidas podem ser realizadas sem causar danos e gere os comandos enviados para os escravos e garra.

A funcionalidade do conjunto de blocos **proteção** e **gestão** é equivalente ao *Interlock* do sistema didático e poderia ser gerida pelo *Interlock*, contudo para garantir o melhor tempo de reação possível foi preferia a implementação local das proteções.

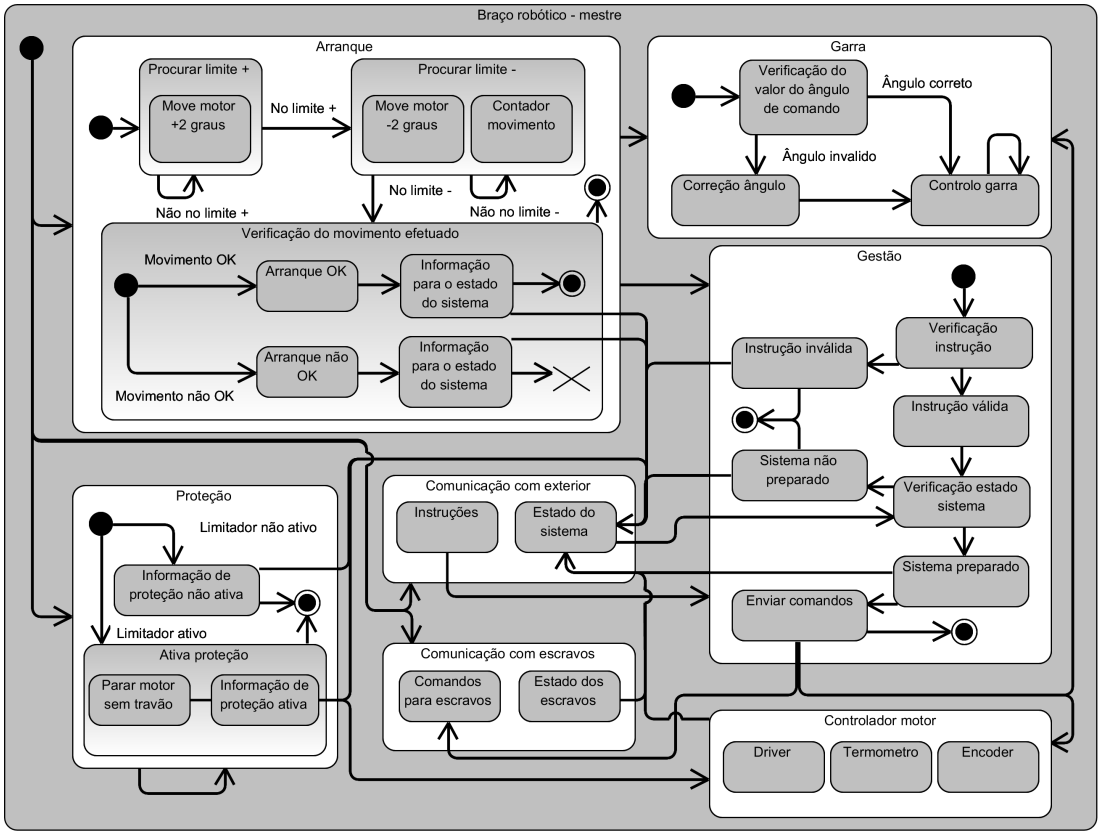


Figura 5.28: *Lógica de funcionamento do controlador mestre do braço robótico*

A arquitetura a implementar no escravo (figura 5.29) difere da do mestre por não possuir os seus dois blocos de comunicação, tendo um único, o **comunicação com o mestre** e não existir bloco **garra**.

- **Comunicação com mestre** é responsável por fornecer informação do estado do escravo e receção das instruções de comando vindas do mestre. Apesar as instruções serem verificadas no mestre antes de serem enviadas para os escravos, elas voltam a ser analisadas no escravo para precaver situações de erro na comunicação do estado do sistema.

O funcionamento e implementação dos blocos é descrito em mais detalhe na próxima secção.

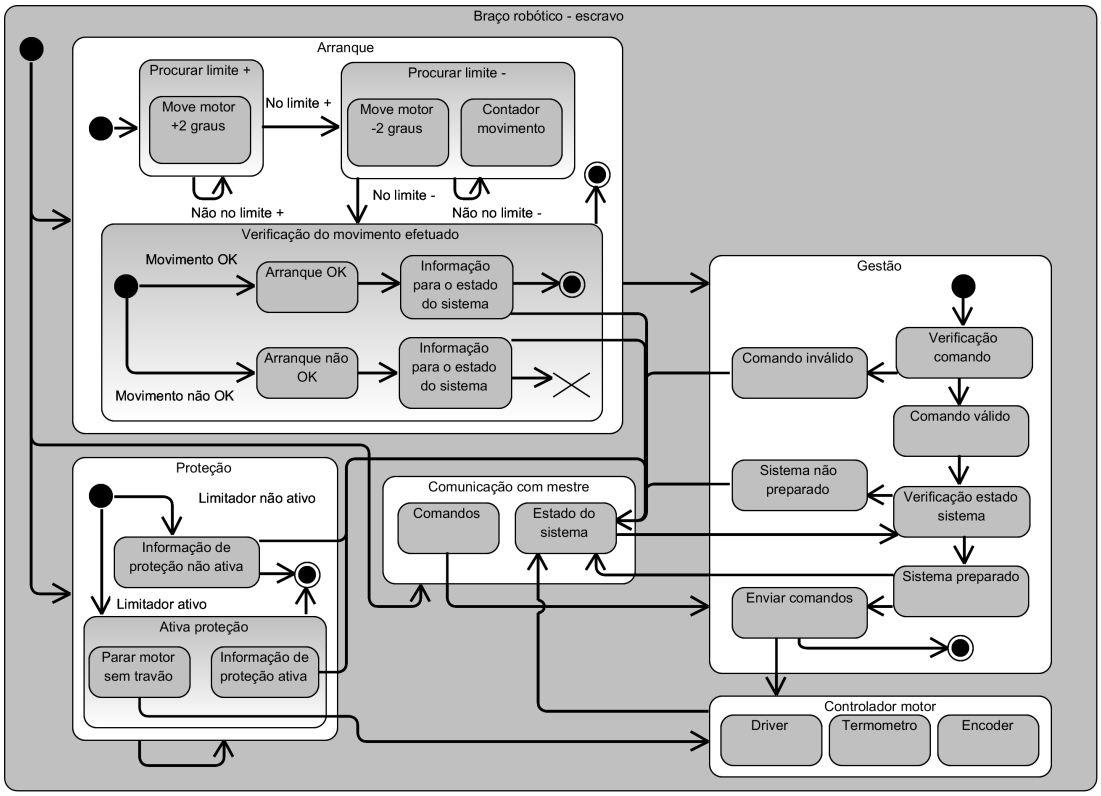


Figura 5.29: *Lógica de funcionamento dos controladores escravo do braço robótico*

5.11.3 Implementação

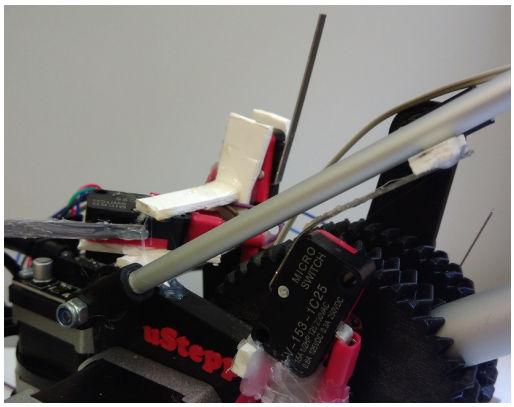
5.11.3.1 Proteção

A deteção dos limites físicos é feita com recurso a fins de curso, instalados nos pontos limite dos eixos (figura 5.30a) e conectados à placa responsável por controlar o motor desse eixo. Um fim de curso adicional instalado no cotovelo do braço (figura 5.30b) permite detetar a colisão entre os eixos responsáveis por variar a altura e alcance do *gripper*, estando conectado a ambas as placas de controlo destes eixos.

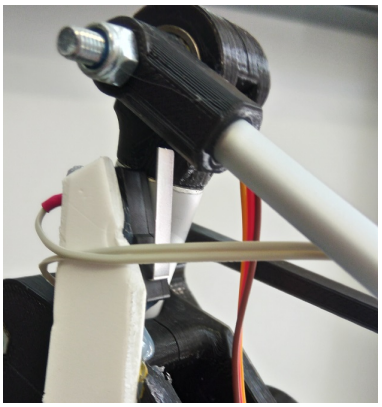
Caso um fim de curso seja atuado o motor é parado sem travão, ficando solto e livre de ser movimentado manualmente.

5.11.3.2 Arranque

De forma a verificar o funcionamento do *hardware* o eixo é levado até o limite positivo e de seguida ao limite negativo, sendo medido o movimento efetuado entre os limites. A comparação do valor medido com uma gama de valores pré-definida, obtida a partir de vários testes, permite detetar que um dos sensores não está a funcionar devidamente, ou, que o movimento sofreu atrasos por deslizamento conforme o valor medido seja respetivamente inferior ou superior aos valores pré-definidos.



(a) Deteção do limite nos eixos do braço



(b) Deteção de colisão no cotovelo

Figura 5.30: fim de curso instalados

Se a verificação detetar algum problema o sistema fica parado no arranque, sendo necessário um *reset* para realizar uma nova verificação. Caso não seja detetado um problema é ativado o bloco de gestão e, no caso do mestre, da garra. O resultado da verificação é colocado numa zona atribuída no estado do sistema.

5.11.3.3 Comunicação com o exterior

A comunicação é feita através de *Modbus RTU* por porta série via *bluetooth*, permitindo a compatibilidade com o *Interlock* e a comunicação sem fios, sendo prevenido o risco de danificar com os movimentos continuados do braço o cabo e/ou porta USB da placa de controlo.

Os registos *Modbus* (tabela 5.4) são divididos em instruções, onde o *Interlock* escreve as instruções e obtém *feedback*, e estado do sistema, onde são disponibilizadas informações do estado do arranque, do mestre e dos escravos.

5.11.3.4 Garra

O ângulo de abertura do *gripper* é controlado por um servomotor, podendo variar a abertura entre os 0 graus para fechado e 160 para aberto.

Nos ângulos limite o servomotor fica em esforço devido ao facto de o *hardware* do *gripper* estar no seu limite mecânico e forçar o ângulo a mudar. De forma a evitar esta situação foi tido como limites a implementar 10 e 150 graus, sendo que os valores comando de abertura recebidos estiverem fora dos limites implementados são ajustados para o limite mais próximo.

Segundo a documentação a função para atualizar o pino de saída do servomotor deve ser constantemente atualizado num intervalo de tempo sempre inferior a 50 milissegundos, pelo que quando não é enviado um novo valor de ângulo para o servomotor é feita uma atualização do anterior, de forma a garantir o seu bom funcionamento.

Tabela 5.4: Registos *Modbus* do braço robótico

<table>
<tr><td>Registo</td><td colspan="3">Função</td></tr>
<tr><td>0</td><td colspan="2">ID</td><td>Identificação do escravo</td></tr>
<tr><td>1</td><td rowspan="2">Intruções</td><td>Receção</td><td>Receção de instruções do exterior,
colocado a 0 depois de processada</td></tr>
<tr><td>2</td><td>Feedback</td><td>0 - Falha na comunicação
1 - Espera
2 - Válida
3 - Não suportada
4 - Não válida</td></tr>
<tr><td colspan="4"></td></tr>
<tr><td>5</td><td rowspan="3">Estado</td><td>Arranque</td><td>0 - Não inicializado
1 - Inicializado com sucesso</td></tr>
<tr><td>6</td><td>Mestre</td><td rowspan="3">0 - Falha de comunicação
1 - Livre 2 - Execução
3 - Proteção ativa</td></tr>
<tr><td>7</td><td>Escravo A</td></tr>
<tr><td>8</td><td>Escravo B</td></tr>
</table>

5.11.3.5 Controlador motor

O controlador é composto por três componentes, a *driver* de potência para o motor de passo, o *encoder* que permite saber a posição do eixo do motor e um termómetro para obter a temperatura na zona da placa por baixo da *driver*. A prevenção de sobreaquecimento não é implementada pelo fabricante, tendo de ser garantida pelo programa desenvolvido.

A informação da temperatura da *driver* e ângulo do motor são ciclicamente atualizados no estado do sistema e as comandos de movimento do motor são executados sempre que recebidos.

5.11.3.6 Comunicação com escravos

O microcontrolador em que as placas são baseadas apenas possui uma porta série nativa, pelo que para o mestre comunicar com os escravos é necessário recorrer a portas série por *software*. O funcionamento de múltiplas portas série por *software* para transmissão é semelhante a uma nativa, contudo para receção é necessária a seleção prévia da porta, levando a atrasos significativos na execução do código.

Sendo necessário no total três portas série, uma para para comunicação com o exterior e duas para comunicação com os escravos, foram implementadas duas portas série por *software*, sendo as portas por *software* apenas utilizadas para transmitir comandos do mestre para os escravos. As respostas dos escravos são obtidas por um conjunto de sinais booleanos nas suas saídas, ligados a entradas do mestre que dão informações do seu estado como livre, em execução ou proteção ativa.

5.11.3.7 Gestão

Ao ser recebida uma nova instrução na **comunicação com o exterior** este bloco é ativado, verificando se a instrução pode ou não ser executada, numa primeira fase por verificação se é uma instrução suportada e depois se é válida ou não dado o estado atual do braço, sendo descartada se

a sua execução poder causar danos ao braço ou envolvência. Se for possível a executar a instrução os devidos comandos são enviados para os escravos e/ou motor e/ou garra.

Devido a uma avaria, descrita nos resultados obtidos, a implementação deste bloco não foi finalizada.

5.11.4 Resultados obtidos

Durante o desenvolvimento foram detetados vários *bugs* no funcionamento do *uStepper Robot Arm*, derivados tanto por *hardware* como pela biblioteca do fabricante. Alguns dos *bugs* foram pinos não indicados como indisponíveis levavam a placa a não inicializar quando utilizados pelo código do utilizador. O controlo do motor com recurso a funções de malha fechada, que utilizam o *encoder*, levaram a comportamentos anormais como o motor rodar a baixa velocidade ou rodar livremente. Também foram encontradas limitações no funcionamento do *gripper* que além da situação descrita na secção 5.11.3.4, ao ser ajustado para certos ângulos de abertura o *gripper* abria e fechava na totalidades algumas vezes até estabilizar no ângulo correto.

As placas de controlo do braço também sofreram avarias. O C.I. de conversão USB para porta série, necessário para programar o microcontrolador, avariou em duas das placas *ustepper* ao longo do desenvolvimento, sem motivo aparente, tendo que se recorrer à programação por *In-system programming* (*ISP*). Apesar de o restante *hardware* estar funcional, e se conseguir programar as placas, a falta de comunicação por porta série não permitia fazer a depuração do funcionamento da placa por recurso a mensagens por terminal.

Uma avaria posterior ocorreu ao ligar a alimentação do braço, tendo um C.I. da *driver* de motor entrado em curto-circuito inviabilizando a utilização da placa. Durante esta fase foi utilizado um Arduino Uno como mestre para ser possível continuar o desenvolvimento, sendo ignorado o movimento do eixo controlado pelo mestre.

Após contacto com o fabricante do braço a reportar a avaria, houve um processo de ativação de garantia que culminou com o envio das três placas para análise. Tal implicou a suspensão do desenvolvimento desta parte da implementação. A análise do fabricante detetou defeitos de fabrico nas placas, tendo sido recebido um novo conjunto de placas de controlo, mas que não chegou a tempo de permitir o retorno do desenvolvimento da implementação.

Os testes ao braço mostraram ter capacidade de mover objetos com peso semelhante às peças do sistema didático com uma boa precisão. É por isso esperado que com uma garra adequada consiga transportar as peças devidamente, contudo a fragilidade das placas de controlo e a possibilidade de poderem existir *bugs* adicionais na biblioteca, não permitem que a solução de desenvolver uma célula baseada neste braço robótico seja viável.

Capítulo 6

Conclusões e trabalho futuro

O objetivo principal deste trabalho era a implementação de soluções propostas para melhoria do sistema didático, tendo sido validado o funcionamento de todas as propostas.

As soluções referentes à base computacional, *Interlock* e identificação das peças por código de barras foram implementadas, garantindo dessa forma as funcionalidades existentes anteriormente.

As propostas referentes à correção da limitação mecânica da célula de montagem e ao programa de demonstração inserido na base computacional também foram implementadas na totalidade.

Os painéis de informação e configuração do sistema didático também foram implementados, permitindo conhecer o estado do sistema de forma rápida e direta, assim como fazer algumas configurações. No entanto, as funcionalidades disponibilizadas pelo *Codesys* permitem realizar muitas outras funções, que não foram exploradas, como a configuração de parâmetros do código do *softPLC*, útil, por exemplo, para ajustar tempos de atuação das proteções.

O funcionamento da expansão da identificação das peças por *RFID* e informação do estado de ocupação do armazém foram validados, sendo a trabalhar no futuro a sua implementação física no sistema didático. Também é deixado para trabalho posterior o controlo dos *I/O Expander* diretamente pelo *Codesys*.

O desenvolvimento das máquinas de simulação de processos para a célula de montagem não foi possível de realizar com o *hardware* idealizado, tendo esta solução ainda de ser desenvolvida segundo a proposta.

A opção escolhida como solução para o desenvolvimento de uma nova célula baseada num braço robótico mostrou não ser fiável devido às placas de controlo de motores utilizadas. Visto o *uStepper Robot Arm* ser baseado em motores de passo a passo é possível o seu controlo ser feito por outras placas de controlo, pelo que fica para mais tarde o estudo de opções de controlo alternativas às placas originais do braço robótico e a implementação da solução com a nova opção de controlo.

Para trabalho futuro é também deixado o desenvolvimento de um sistema de identificação e seguimento das peças a partir de visão e a avaliação automática na base computacional de trabalhos laboratoriais.

Referências

[1] Avago Technologies. *APDS-9104 Integrated Reflective Sensor*, 1 2007.

[2] André Sarmento Barbosa. Github - andresarmento/modbus-arduino: A library that allows your arduino to communicate via modbus protocol, acting as a slave (master in development). supports serial (rs-232, rs-485) and ip via ethernet (modbus ip). https://github.com/andresarmento/modbus-arduino, 2015. [Online; acedido 10 Março 2018].

[3] BeagleBoard.org. Beagleboard.org - community supported open hardware computers for making. https://beagleboard.org/, 2018. [Online; acedido 3 Janeiro 2018].

[4] Beremiz. Home page of beremiz. https://beremiz.org/, 2017. [Online; acedido 10 Dezembro 2017].

[5] J. R. Caldas Pinto e J. M. G. Sä da Costa. Virtual and remote laboratories for industrial automation e-learning. *IFAC Proceedings Volumes*, 46(17):286–290, 2013.

[6] Circuits@Home. Usb host shield hardware manual « circuits@home. https://www.circuitsathome.com/usb-host-shield-hardware-manual/, 2018. [Online; acedido 22 Março 2018].

[7] Paulo Costa. Feup - paulo gomes da costa. https://sigarra.up.pt/feup/pt/func_geral.formview?p_codigo=211795, 2018. [Online; acedido 2 Março 2018].

[8] OWI Inc. dba: Robotikits^{TM} Direct. Robotic arm edge. http://www.owirobot.com/robotic-arm-edge-1/, 2017. [Online; acedido 5 Dezembro 2017].

[9] Alberto De Toni e Stefano Tonchia. Manufacturing flexibility: a literature review. *International journal of production research*, 36(6):1587–1617, 1998.

[10] Schneider Electric. *Advantys STB Standard Ethernet Modbus TCP/IP Network Interface Module Applications Guide*. Schneider Electric, 8 2013.

[11] Schneider Electric. *IP 20 distributed inputs/outputs Modicon STB*. Schneider Electric, 5 2013.

[12] Schneider Electric. Distributed i/o - modicon stb | schneider electric. https://www.schneider-electric.com/en/product-range/606-modicon-stb, 2017. [Online; acedido 10 Dezembro 2017].

[13] Schneider Electric. What are the function codes of the modbus protocol used by io scanning in tcp-ip? https://www.schneider-electric.co.uk/en/faqs/FA32793/, 2018. [Online; acedido 12 Março 2018].

[14] Feig Eletronic. Iso15693 proximity desktop reader - id isc.pr101 - identification - products - feig electronic. https://www.feig.de/en/products/identification/product/id-iscpr101/, 2018. [Online; acedido 8 Janeiro 2018].

[15] Famic. Famic technologies - automation software - automation studio - grafcet editor - opc plc sfc sps hmi - hydraulic pneumatic simulator - workflow. https://www.famictech.com/, 2018. [Online; acedido 21 Janeiro 2018].

[16] Lyle D. Feisel e Albert J. Rosa. The role of the laboratory in undergraduate engineering education. *Journal of Engineering Education*, 94(1):121–130, 2005.

[17] Festo. Home | festo portugal. https://www.festo.com/cms/pt_pt/index.htm, 2018. [Online; acedido 15 Janeiro 2018].

[18] Raspberry Pi Foundation. Camera module - raspberry pi documentation. https://www.raspberrypi.org/documentation/usage/camera/README.md, 2018. [Online; acedido 15 Maio 2018].

[19] Raspberry Pi Foundation. Raspberry pi - teach, learn, and make with raspberry pi. https://www.raspberrypi.org/, 2018. [Online; acedido 3 Janeiro 2018].

[20] Raspberry Pi Foundation. Raspberry pi forums - index page. https://www.raspberrypi.org/forums/, 2018. [Online; acedido 11 Março 2018].

[21] Raspberry Pi Foundation. Using a standard usb webcam - raspberry pi documentation. https://www.raspberrypi.org/documentation/usage/webcams/, 2018. [Online; acedido 15 Maio 2018].

[22] Real Games. Its plc | plc training software by real games. https://realgames.co/its-plc/, 2017. [Online; acedido 5 Dezembro 2017].

[23] Real Games. Next-gen plc training software with factory i/o. https://factoryio.com/, 2017. [Online; acedido 5 Dezembro 2017].

[24] 3S-Smart Software Solutions GmbH. How to connect serial devices ? - codesys - the iec 61131-3 automation software. https://forum.codesys.com/viewtopic.php?f=21&t=5699&sid=dea83180e8f5b6a93d988849d490f2fd, 2014. [Online; acedido 11 Março 2018].

[25] 3S-Smart Software Solutions GmbH. Codesys store - codesys control for beaglebone sl. https://store.codesys.com/codesys-control-for-beaglebone-sl.html, 2017. [Online; acedido 10 Dezembro 2017].

[26] 3S-Smart Software Solutions GmbH. Codesys store - codesys control for raspberry pi sl. https://store.codesys.com/codesys-control-for-raspberry-pi-sl.html, 2017. [Online; acedido 10 Dezembro 2017].

[27] 3S-Smart Software Solutions GmbH. Home - codesys. https://www.codesys.com/, 2017. [Online; acedido 10 Dezembro 2017].

[28] Fischertechnik GmbH. Simulating - fischertechnik. https://www.fischertechnik.de/en/simulating, 2017. [Online; acedido 20 Outubro 2017].

[29] Staudinger GmbH. Simulation, fischertechnik, models, compatible aluminiumprofiles, plc-starter-kit. http://www.staudinger-est.de/en/simulation/, 2017. [Online; acedido 20 Outubro 2017].

[30] Andreas Größler. Don't let history repeat itself—methodological issues concerning the use of simulators in teaching and experimentation. *System Dynamics Review*, 20(3):263–274, 2004.

[31] I. Gustavsson, K. Nilsson, J. Zackrisson, J. Garcia-Zubia, U. Hernandez-Jayo, A. Nafalski, Z. Nedic, O. Gol, J. Machotka, M. I. Pettersson, T. Lago, e L. Hakansson. On objectives of instructional laboratories, individual assessment, and use of collaborative remote laboratories. *IEEE Transactions on Learning Technologies*, 2(4):263–274, 2009.

[32] IEC. Iec 61131-3:2013 | iec webstore | water automation, water management, smart city. https://webstore.iec.ch/publication/4552, 2018. [Online; acedido 20 Janeiro 2018].

[33] OWI Inc. *OWI-535 Robotic Arm Edge Manual*. 2008.

[34] RobotShop inc. Lynxmotion - robotic arms. http://www.lynxmotion.com/c-27-robotic-arms.aspx, 2017. [Online; acedido 6 Dezembro 2017].

[35] InCircuit. Icnova ap7000 base - incircuit. https://wiki.in-circuit.de/index.php5?title=ICnova_AP7000_Base, 2017. [Online; acedido 2 Dezembro 2017].

[36] Adafruit Industries. Adafruit 16-channel 12-bit pwm/servo driver - i2c interface [pca9685]. https://www.adafruit.com/product/815, 2007. [Online; acedido 15 Dezembro 2017].

[37] IRAI. Irai, automation simulation, plc programming, pneumatic simulation. http://www.iraifrance.com/. [Online; acedido 5 Dezembro 2017].

[38] Identification cards – Contactless integrated circuit cards – Proximity cards – Part 1: Physical characteristics. Standard, International Organization for Standardization, Geneva, CH, 2016.

[39] Identification cards – Contactless integrated circuit cards – Proximity cards – Part 2: Radio frequency power and signal interface. Standard, International Organization for Standardization, Geneva, CH, 2016.

[40] Identification cards – Contactless integrated circuit cards – Proximity cards – Part 3: Initialization and anticollision. Standard, International Organization for Standardization, Geneva, CH, 2016.

[41] Identification cards – Contactless integrated circuit cards – Proximity cards – Part 4: Transmission protocol. Standard, International Organization for Standardization, Geneva, CH, 2016.

[42] Identification cards – Contactless integrated circuit cards – Vicinity cards – Part 2: Air interface and initialization. Standard, International Organization for Standardization, Geneva, CH, 2006.

[43] Identification cards – Contactless integrated circuit cards – Vicinity cards – Part 3: Anticollision and transmission protocol. Standard, International Organization for Standardization, Geneva, CH, 2009.

[44] Identification cards – Contactless integrated circuit cards – Vicinity cards – Part 1: Physical characteristics. Standard, International Organization for Standardization, Geneva, CH, 2000.

[45] ON Development IVS. *uStepper product description Rev. B*. 2015.

[46] ON Development IVS. *uStepper - Robot Arm Assembly instructions*. 2016.

[47] ON Development IVS. ustepper. http://www.ustepper.com/index/, 2017. [Online; acedido 5 Dezembro 2017].

[48] ON Development IVS. ustepper · github. https://github.com/uStepper/, 2018. [Online; acedido 1 Março 2018].

[49] Bjarte Johansen. Github - miguelbalboa/rfid: Arduino rfid library for mfrc522. https://github.com/miguelbalboa/rfid, 2018. [Online; acedido 2 Fevereiro 2018].

[50] Oleg Mazurov, Alexei Glushchenko, Kristian Lauszus, Andrew Kroll, e Yuuichi Akagaw. Github - felis/usb_host_shield_2.0: Revision 2.0 of usb host library for arduino. https://github.com/felis/USB_Host_Shield_2.0, 2018. [Online; acedido 22 Março 2018].

[51] M. J. G. C. Mendes e L. Martins. An internet remote laboratory to teach industrial automation. Em *2014 7th International Conference on Human System Interactions (HSI)*, páginas 144–149.

[52] Microchip Technology Inc. *MCP23017/MCP23S17 16-Bit I/O Expander with Serial Interface*, 7 2016. Rev. C.

[53] Lucas Nülle. Lucas nülle - lucas-nuelle training systems for vocational training and didactic. https://www.lucas-nuelle.com, 2018. [Online; acedido 20 Janeiro 2018].

[54] NXP Semiconductors. *PCA9685 16-channel, 12-bit PWM Fm+ I^2C-bus LED controller*, 4 2015. Rev. 4.

[55] NXP Semiconductors. *MFRC522 Standard performance MIFARE and NTAG frontend*, 4 2016. Rev. 3.9.

[56] NXP Semiconductors. *PN5180A0xx/C1/C2 High-performance multi-protocol full NFC Forum-compliant frontend*, 7 2017. Rev. 3.3.

[57] OpenPLC. The openplc project. http://www.openplcproject.com/, 2017. [Online; acedido 8 Dezembro 2017].

[58] OpenPLC. The openplc project | getting started - rpi. http://www.openplcproject.com/getting-started-rpi, 2017. [Online; acedido 8 Dezembro 2017].

[59] OpenPLC. Openplc forum. https://openplc.discussion.community/, 2018. [Online; acedido 12 Fevereiro 2018].

[60] M. M. Pandini, A. Diogo Spacek, J. M. Neto, e O. H. Ando Junior. Design of a didatic workbench of industrial automation systems for engineering education. *IEEE Latin America Transactions*, 15(8):1384–1391, 2017.

[61] Sébastien Parent-Charette. How to choose a lynxmotion robotic arm - robotshop blog. https://www.robotshop.com/blog/en/lynxmotion-robotic-arms-15845, 2015. [Online; acedido 6 Dezembro 2017].

[62] K. Phillips, B. Gruszka, J. Carroll, M. Bauer, e O. Maharaj. Connecting industrial automation software to a discrete manufacturing plant model for research and education. Em *2013 Africon*, páginas 1–5.

[63] proconX Pty Ltd. Modbus slave simulator and test tool. http://www.modbusdriver.com/diagslave.html, 2018. [Online; acedido 10 Março 2018].

[64] Samuel. Github - smarmengol/modbus-master-slave-for-arduino: Modbus master-slave library for arduino. https://github.com/smarmengol/Modbus-Master-Slave-for-Arduino, 2016. [Online; acedido 10 Março 2018].

[65] Witte Software. Modbus master simulator. http://www.modbustools.com/modbus_poll.html, 2018. [Online; acedido 10 Março 2018].

[66] Witte Software. Modbus slave simulator. http://www.modbustools.com/modbus_slave.html, 2018. [Online; acedido 10 Março 2018].

[67] Armando Jorge Sousa. Feupautom - armando jorge sousa. https://web.fe.up.pt/ asousa/wiki/doku.php?id=proj:feupautom, 2017. [Online; acedido 20 Dezembro 2017].

[68] OpenCV team. Opencv library. https://opencv.org/, 2018. [Online; acedido 15 Maio 2018].

[69] Texas Instruments. *CD405xB CMOS Single 8-Channel Analog Multiplexer/Demultiplexer With Logic-Level Conversion*, 9 2017. Rev. I.

[70] TT Electronics. *NPN Silicon Phototransistor OP505, OP506, OP535 & OP705 Series*, 08 2016. Rev. A.

[71] UDOO. All-you-need mini pc android + linux + arduino | udoo. https://www.udoo.org/, 2018. [Online; acedido 3 Janeiro 2018].

[72] Vishay Semiconductors. *TSAL4400 High Power Infrared Emitting Diode, 940 nm, GaAlAs, MQW*, 11 2016. Rev. 1.9.

[73] Jason Vreeland. Google code archive - long-term storage for google code project hosting. https://code.google.com/archive/p/arduino-modbus-slave/downloads, 2012. [Online; acedido 10 Março 2018].